Estimation of Unsteady Nonuniform Heating Rates from Surface Temperature Measurements

by

Greg Walker, B.S., M.S

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Mechanical Engineering

Elaine Scott, Chair
Thomas Diller
Tao Lin
Wing Ng
Brian Vick

December 1997

Blacksburg, Virginia

Estimation of Unsteady Nonuniform Heating Rates from Surface Temperature Measurements

Approved by
Dissertation Committee:

_____

_____

_____

_____

_____

To my wife, Cynthia

Acknowledgments

I would like to thank NASA Langley Research Center (LaRC) for awarding a *Graduate Student Research Program* Fellowship (NGT–51249) to make this work possible, and Dr. Robert Nowak, Research Scientist at NASA LaRC, for his guidance and willingness to help with the data presented here. Some of the really cool pictures were provided by the Aerothermodynamics Branch at NASA LaRC as well.

Among my colleagues and friends that deserve mention are Tom Leitch for stellar illustrations and Dave Coe for commiseration. All the folks in the Heat Transfer Lab helped maintain my sanity via "Lab Night Out" (and what not). Further thanks go to Shari Feth and Paul Tidwell for helping me focus on non-academic activities. I would also like to thank Dr. R. L. West for allowing me to pound his computing facilities so I could complete this work. Without his professional courtesy, my dissertation would have suffered tremendously.

Thanks also go to Elaine Scott, my advisor, for the lessons on diplomacy and for bailing me out of some tight situations. She also always seemed to find the funding for me to advance my career at conferences and seminars. I know this was no small task.

Finally, I would like to thank my family, especially my wife. I think this degree stressed her more than it did me.

Greg Walker

Virginia Polytechnic Institute and State University
December 1997

Estimation of Unsteady Nonuniform Heating Rates from Surface Temperature Measurements

Greg Walker, B.S., M.S
Virginia Polytechnic Institute and State University, 1997
Supervisor: Elaine P. Scott

Abstract

Shock wave interactions such as those that occur during atmospheric re-entry, can produce extreme thermal loads on aerospace structures. These interactions are reproduced experimentally in hypersonic wind tunnels to study how the flow structures relate to the deleterious heat fluxes. In these studies, localized fluid jets created by shock interactions impinge on a test cylinder, where the temperature due to the heat flux is measured. These measurements are used to estimate the heat flux on the surface as a result of the shock interactions. The nature of the incident flux usually involves dynamic transients and severe nonuniformities. Finding this boundary flux from discrete unsteady temperature measurements is characterized by instabilities in the solution. The purpose of this work is to evaluate existing methodologies for the determination of the unsteady heat flux and to introduce a new approach based on an inverse technique. The performance of these methods was measured first in terms of accuracy and their ability to handle inherently "unstable" or highly dynamic data such as step fluxes and high frequency oscillating fluxes. Then the method was expanded to estimate unsteady and nonuniform heat fluxes. The inverse methods proved to be the most accurate and stable of the methods examined, with the proposed method being preferable.

Contents

List of Tables

List of Figures

Nomenclature

c_p	specific heat (kJ/kg K)
G	Green's function
$\mathbf{H}$	order of regularization matrix (dimensionless)
k	thermal conductivity (W/m K)
M	Mach number
p	Laplace transform function
q	integrated heat flux (W sec/m^2)
$\dot{q}$	heat flux (W/m^2)
S	objective function (K^2)
t	time (sec)
T	calculated temperature (K)
x	space coordinate (m)
X	sensitivity coefficient (K m^2/W)
$\mathbf{X}$	Sensitivity Matrix (K m^2/W)
Y	measured temperature (K)
κ	thermal diffusivity (m^2/sec)
α	regularization parameter (K^{2}m^4/W^2)
β'	variable property correction factor (1/K)
Δx	distance between nodes (m)
η	transformed spatial coordinate (dimensionless)
ϕ	Kirchoff temperature (K)
Ψ^{-1}	matrix of measurement variances (1/K^2)
ρ	density (kg/m^3)

τ	transformed time coordinate
ξ	variable of integration

Subscripts and superscripts

i	spatial index
j	temporal index
n	time step
o	nominal value
s	surface value
$^-$	transform variable

Chapter 1

Introduction

Extreme thermal environments are found during high speed flight and atmospheric re–entry. Such environments can be generated from shock interactions created by high speed aerial maneuvers. As aerospace vehicles are designed to travel faster, the heating rate imposed on the craft as a result of the interactions become dangerously large. Due to the deleterious effects of shock interaction on supersonic craft, scientists have been studying the phenomenon in an attempt to improve designs that mitigate the potential for destruction.

Tests designed to uncover the nature of shock interactions frequently involve finding heat fluxes incident on a hypersonic wind tunnel model that is subjected to extreme heating. These unsteady, nonuniform heating rates must be resolved from temperature measurements taken from the surface of the model. Because of the difficulty of the data reduction, this work focuses on the solutions to this problem.

The overall goal of this research, then, is to develop a methodology suitable for data reduction of measurements such as those made at hypersonic facilities. The methodologies developed here should also be able to expand the flexibility of future experiments. Currently, experimental designs are constrained by the limitations of data reduction techniques. With advanced techniques, the potential for enhanced experiments is augmented.

To these ends, the nature of shock-shock interactions will be discussed in the context of determining the resultant heating loads on aerospace structures. Furthermore, descriptions of the experiments and characteristics of the measurements will be presented. Having defined the problem, several commonly used solution techniques will be presented along with the proposed methodology. Finally, the solution methodologies will be evaluated using

test cases with simulated data and using experimental surface temperature measurements.

1.1 Motivation

Extremely high heat fluxes can be created by shock interactions during atmospheric re–entry and high speed flight of aerospace vehicles. The fluxes can be large enough to damage vehicles by imposing tremendous thermal stresses on the structures.

Perhaps the first evidence of the pernicious effects of shock interactions appeared in late 1960's during an X-15 flight designed to study the performance of a ram-jet attached to the wing of the plane. The test article was held in place by a pylon extending toward the intake of the jet and attached to the underside of the X-15's wing. The high speed of the flight resulted in shocks being shed from both the nose of the test article and the leading edge of the X-15 wing as described by Burcham, Jr. and Nugent (1970). The intersection of two shocks produced a hypersonic flow pattern that impinged the strut. Watts (1968a) describes and analyzes the extreme heating and ultimate failure due to irreparable damage to the support member (Fig. 1.1). Fortunately the test article was not lost until the mission had been completed. (It is rumored that the ram-jet fell off the plane as it rolled to a stop after landing.)

Today's aerospace vehicles are designed to be faster and are required to withstand larger heat fluxes than those in the 1970's. For example, the space shuttle routinely experiences some level of damage due to shock interaction heating. Edney (1970) outlines the dangers of shock interactions related to shuttle launches and re–entry. Potential damage locations are depicted in Fig. 1.2. As scientists begin to design the next generation of spacecraft, such as the X–33, they must have a thorough understanding of this phenomenon and the potential heating rates involved with the new designs. Through experimentation, the effects of shock interaction heating can be measured. The interpretation of these measurements, then, leads to a greater understanding of the physics.

1.2 History of Shock Interaction Studies

As early as 1967, Hiers and Loubsky (1967) examined the heating effects of shock interactions on cylinders with different sweep angles. This configuration was designed to model a bow shock being shed from the nose of a plane intersecting a compression shock created by

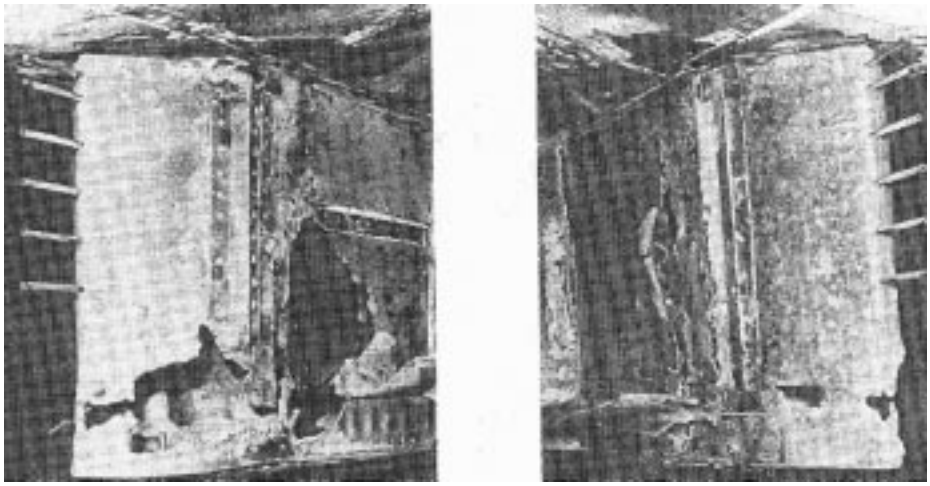

Fig. 1.1 Damage to the X–15 strut due to shock interactions (Holden et al., 1991).

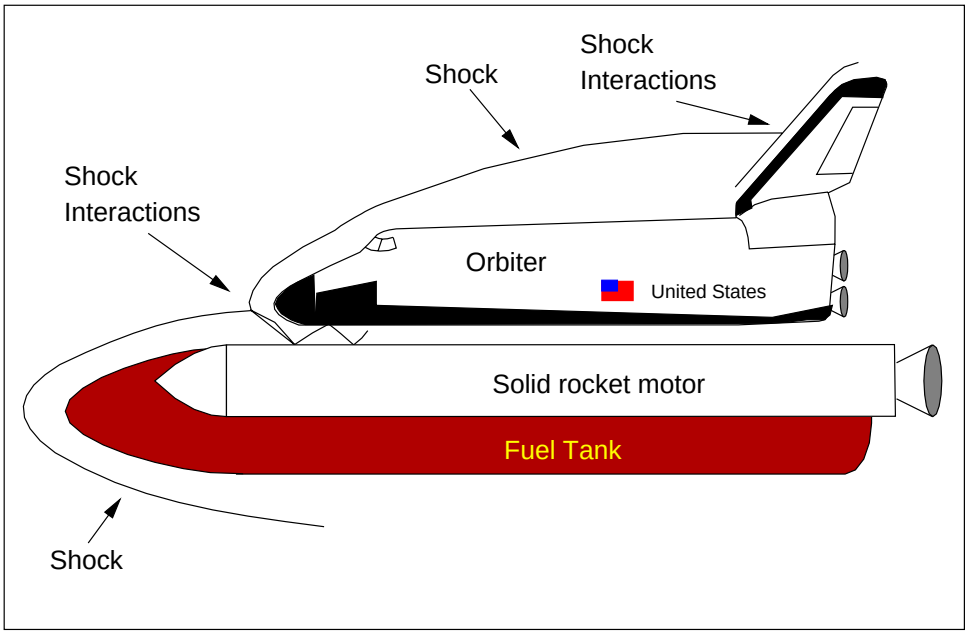

Fig. 1.2 Locations of potential damage due to shock interaction on the Space Shuttle.

a control surface such as a tail, wing or control flap. Each scenario was known to generate extreme heating on real aircraft. There have been instances when heating rates for shock interaction conditions have been reported to be ten times greater than stagnation flow heating rates. However, a normal increase in heating rate due to shock interactions is three to four times that of stagnation flows. At the same time Edney (1968a) was performing similar studies on blunt bodies and classifying types of shock interactions. This experimental approach employed a simple planar shock (created by an upstream wedge) and compression shock created by a change in the angle of the surface to the free-stream flow. This setup was designed to test scientists' fundamental understanding of shock-shock interactions and build a base of knowledge for future experimentation and evaluation. This work has become the canonical reference for those who study these phenomenon. The results of both approaches were unanimous that the heating was severe, however, the mechanism for the heating was largely not understood despite additional experimental work performed by Edney (1968b) to more completely characterize shock interactions.

Shock interaction studies were motivated by damage to high speed aircraft, namely the X-15. Watts (1968a) outlined some of the problems with the experimental craft that were a direct result of shock interaction heating. Locations of melting and visible damage to the ablative heat shield were noted as a result of the Mach 6 flight. Watts (1968b) also noted damage to the high speed aircraft near and on protuberances on the plane. It was surmised that this damage was also a result of shock interaction heating. More specifically, Burcham, Jr. and Nugent (1970) noted a particular experiment on the X-15 designed to study the effects of shock interaction heating on a dummy ram-jet attached to the underside of the craft as described in Section 1.1. The ram-jet was to be used to boost the plane's thrust at high altitudes and speeds. The experiment was to determine what, if any, regions would experience failure due to shock interactions. The results of the study suggested further research should be performed before the ram-jet could be put in service safely.

Early research on shock interaction was primarily conducted at NASA Langley Research Center (LaRC) where several scientists worked on developing relationships between flow parameters and heating rates. This type of work was first conducted by Hains and Keyes (1972), and by Keyes and Morris (1972). Because of the complexity of the flows and different situations in which the flows are found, these relations would be found not to apply universally.

Soon, computers were being employed to describe relationships for heating rates. For example, Morris and Keyes (1973) used oblique shock and Prandtl-Meyer expansion relationships to predict the interference pattern and heat transfer rates. Keyes and Hains (1973) then reported that these solutions were not universally applicable because the distance of the shock from the model had to be determined experimentally. Tannehill et al. (1976) attempted a two-dimensional finite difference Navier Stokes solution, but found that the shock could generally not be "captured" by a coarse model. At the time computer technology did not allow adequate resources for a description of the problem. This work, although helpful in creating an understanding of the phenomena of shock interactions, proved that experimentation was still the most valuable tool in estimating heating rates for different situations.

As research continued in the mid-seventies, two facilities emerged as the premiere contributors to hypersonic shock interaction understanding: Calspan Corporation (at SUNY Buffalo) and NASA LaRC in Hampton, Virginia. Holden (1975) at Calspan and Miller (1990) at NASA LaRC describe the numerous on–base test rigs used to perform hypersonic, shock iteration studies. Also at this time, scientists began to examine the capabilities of the facilities where the tests were being run. Furthermore, the test equipment was evaluated for its accuracy and as well as its appropriateness for a given situation (Holden, 1975Miller, 1981). Among the issues considered was whether the response time of the gauges was small enough to react to the short experimental time as well as whether the spatial resolution of the gauges could capture the spatial features of the heating. This type of work helped to solidify the methodologies by which experimental data were obtained and to itemize the limitations of current experimentation.

By the late 1980's, experiments were documenting the effects of shock interaction heating for a wide range of conditions. The experiments were often driven by particular projects and therefore, had a narrow focus. Wieting and Holden (1987) studied the effects of leading edge heating for ranges of Mach numbers comparable to that of the National Aerospace Plane and the X–33. These efforts also focused on two-dimensional flow patterns where the test cylinder is oriented parallel to the incoming shock. The intent of the work was to model cowl lips of rectangular engine inlets. This type of work was continued and extended by Holden et al. (1988) to include Mach numbers in excess of 8. A more complete picture of leading edge cowl lip heating was examined by Glass et al. (1989) when the effects of sweep angle were incorporated into their study. Finally, Holden et al. (1991) generated

large amounts of data for the two-dimensional shock interaction heating for leading edges as a baseline for comparison of other models.

While experiments were being performed, numerical modeling techniques (CFD) began to gain promise and accuracy. (Wieting et al., 1988) enjoyed some of the earliest success due to the increased capacity of computers. The problem of "capturing" a shock numerically could only be solved with fine grids or adaptive gridding techniques, both of which require computing power that was becoming available during the late 1980's. Perhaps the most complete analysis to date was performed by Prabhu (1994) who was able to verify a large variety of shock interaction heating problems using an unstructured mesh. The codes developed were also capable of incorporating real gases which more closely matched the conditions during flight. Note that most tunnel experiments can simplify the analysis by using only ideal gases that do not dissociate during the tests.

It is well understood that hypersonic flight can result in extreme thermal loads on the flight vehicle. Of particular interest is the propulsion system which invariably must be subjected to some level of shock interaction heating. In an effort to design structures to withstand these pernicious environments, researchers have begun to explore actively cooled structural components. Melis et al. (1989) and Gladden and Melis (1994) have performed extensive testing on such components using hot gas facilities with shock interactions.

Another area of recent interest returns to the three-dimensional flow patterns examined earlier. Berry and Nowak (1996) provide a unique insight into the heating rates associated with fin leading edges on vehicles such as the National Aerospace Plane and the replacement shuttle. Such experiments demonstrate the value of experimentation over pure analysis. Despite gains in numerical ability, there still exists a discrepancy between measured data and numerical simulation. Therefore, experiments are continually being improved (Neumann, 1996) to further the knowledge of this area of science that is becoming crucial to our success in space exploration and high speed flight.

Our understanding of shock interactions has been greatly enhanced by (if not completely derived from) experimentation. However, we must be able to interpret quantitatively the measurements that are made during the experiments to gain any useful insight. As equipment and measurement techniques become more precise, the need for reliable, accurate data reduction increases. Thus, the field of data reduction is the primary focus of this research.

1.3 Data Reduction

High heat fluxes such as those which occur in shock interaction experiments are typically evaluated by measuring the temperature on the surface of a model and employing a data reduction scheme to convert these temperatures to a corresponding flux. There will be two specific experiments highlighted in this work. Both experiments examined (Holden et al., 1988Berry and Nowak, 1996) currently use this type of scheme.

The data reduction problem is typically plagued by unstable solutions brought about by inherent mathematical instabilities. This can be demonstrated by examining the relationship between the measured temperature and the surface heat flux. A conventional temperature solution for conduction through a material can be expressed as some integral of the boundary flux.

$$T(x,t) = \int q(t)G(x,t)\,dt \tag{1.1}$$

where $q(t)$ is the boundary flux we are searching for, and $G(t)$ represents a characteristic function such as the Green's function. According to Beck et al. (1985), by attempting to find the kernel of this integral (the boundary flux), we introduce some mathematical uncertainty.

The difficulties of finding a solution to this problem are outlined by Tikhonov and Arsenin (1977) Because the data are sampled at discrete times and locations, the problem does not have a unique solution. This feature alone complicates the solution process. What makes the problem unstable, however, is the fact that we must differentiate the data to obtain a solution. The differentiation of discrete data is inherently unstable as discovered by Ehrich (1954). He attempted to condition his data before the differentiation took place to placate the unstable tendencies of his solution method. Although this was a satisfactory solution for a problem with relatively benign measurements, it is not preferable for dynamic data, because information tends to get lost (Trucco and Tamango, 1990). Problems such as these are considered mathematically ill–posed, and their solutions are said to be sensitive. This means small changes in a measurement can result in large changes in the solution.

This complication is compounded because there is usually some level of uncertainty in the data. The type of uncertainty can range from incomplete property data to an imprecise experimental setup. For example, when composite or advanced materials are used in the model, the conduction properties may not be well established. Also, the process of applying

sensors may have some error associated with the precision of the location of the sensor. While these factors contribute to the uncertainty in the model, perhaps the most influential uncertainty is measurement noise. Introduced by the data acquisition system, normally distributed random errors are introduced into the digital signal by the electronics. This noise constitutes a small change in the measurement which, as a result, can significantly affect the solution. Additional uncertainties, and therefore sources of solution degradation, include unknown thermal properties, approximations in numerical solutions, bias in the temperature measurements and lack of confidence in parameters associated with the modeling process. Because of the already large sensitivity of the solution to experimental parameters, any uncertainty will be amplified. These effects become even more significant when dealing with high heat fluxes such as those associated with shock interactions because the gradients are extremely steep and, therefore, more difficult to resolve.

All methods that have been used and the methods that will be proposed in this work have been classified into three groups according to their approach to the solution. Class 1 techniques are identified by analytic solutions to the conduction problem, and are therefore called *Forward Analytic*. For example, the temperature response to a boundary condition can be found using one of many standard solution techniques given in Carslaw and Jaeger (1959). The heat flux is then algebraically recovered from the temperature solution. The second class of techniques are numerical in nature, so they are called *Forward Numeric*. For Class 2, the measured data are used as a boundary condition in a numerical solution to the conduction problem. Once the distribution is known, Fourier's law is used to recover the incident flux. Finally, the third class is called *Inverse*, because the solution methodology attempts to solve the problem by calculating that quantity which is known (the surface temperature), and comparing the calculated data to the measured data. The specific methodologies used in all three classes will be described in greater detail in subsequent chapters.

1.4 Lateral Conduction

Because the nature of the impingement jets is unsteady and highly localized, the methods used to estimate the fluxes should be able to resolve spatial gradients, as well as unsteadiness, inherent in the heating rates. Most data reduction approaches described above and in the literature focus on the heating rate of a single measurement location, and use a

one-dimensional semi-infinite slab assumption. However, this one-dimensional assumption will be shown to be a poor approximation in some cases.

As a quick example of the fallacy of the one-dimensional assumption, suppose a heat flux is being applied to a surface in a step fashion such that there is some area that sees a zero flux and another region sees some nominal flux. For argument's sake, also assume that this spatially dependent flux is constant in time. We know that the heat being added will diffuse into the non-heated region causing the temperature to rise where there is no applied flux. This scenario is demonstrated by examining the two-dimensional temperature distribution resulting from this flux after some time (as in Fig. 1.3). A one-dimensional data reduction routine, will result in a non-zero flux estimate for any sensor that sees a temperature change. However, a two-dimensional scheme, if formulated correctly, should be able to account for the lateral conduction effects and provide a more accurate solution.

It is expected that the lateral conduction effects are significant for our problem (Gladden and Melis, 1993), but a solution method has not been suggested. Buttsworth and Jones (1997 and Chadwick (1997) acknowledged the existence of lateral conduction effects. However, the nonuniform heating was due to the curvature of a model, and not the nonuniformity of the applied heat flux. Lateral conduction effects resulting from non flat models were further examined (Buttsworth and Jones, 1997a) using a high spatial resolution measurement technique. These methods, however, were not applied to a nonuniform flux situation.

1.5 Objectives

There are three primary objectives of this work. Firstly, we want to establish when and the reason why conventional heat flux estimation techniques may not provide accurate estimates. Secondly, we want to introduce the inverse approach as a more accurate and stable mechanism for estimating heat fluxes especially in situations that can not be handled adequately by traditional methods. Finally, we want to demonstrate the ability of the inverse procedures to accurately characterize multi-dimensional, unsteady high heat fluxes found in shock interaction experiments with greater accuracy.

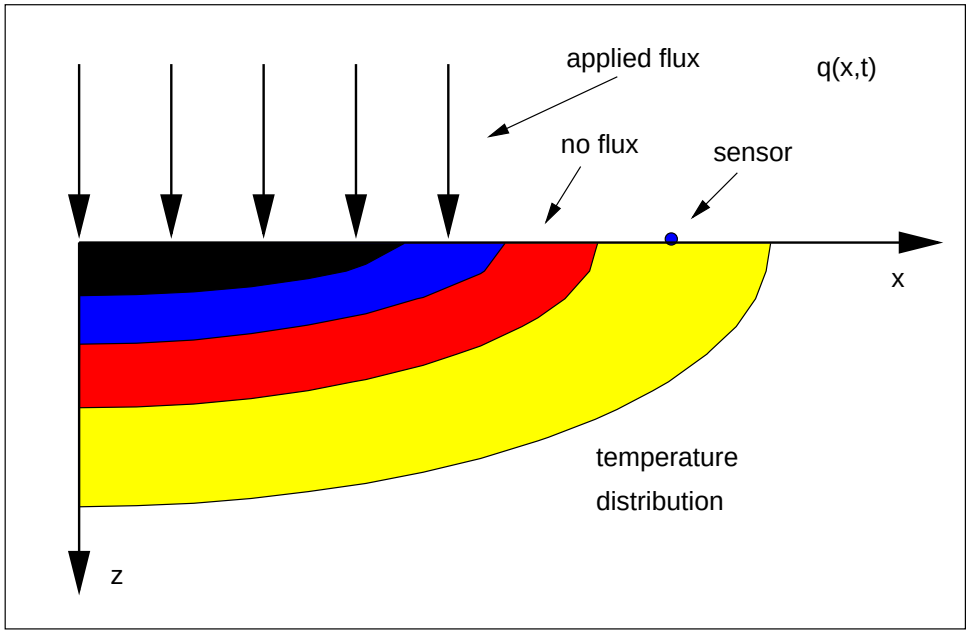

Fig. 1.3 The two-dimensional temperature distribution due to a spatial step flux.

1.6 Terminology

Since there will be several different levels of data to be considered when evaluating solution methodologies, Fig. 1.4 is provided as a map for the conversion process between types of data.

In reality, we have some heat flux as a function of space and time that is prescribed by the flow patterns and physical nature, called the "real flux"; this can not be known exactly. This flux produces a temperature history on the surface of the model that is measured (hence "measured temperature"). The estimator is a method by which the discrete temperature measurements are converted to a representation of the function that is the real flux. From the estimated flux, we can then obtain a temperature solution through a direct conduction solution methodology. How well the estimated flux represents the real flux should be related to how well the temperature solution matches the measured temperature through what we call the residual.

To establish more completely the performance of the estimator we turn to the second scenario in Fig. 1.4. If we start with a predetermined flux, temperature measurements can be generated which we call simulated data. Producing estimated fluxes and tempera-

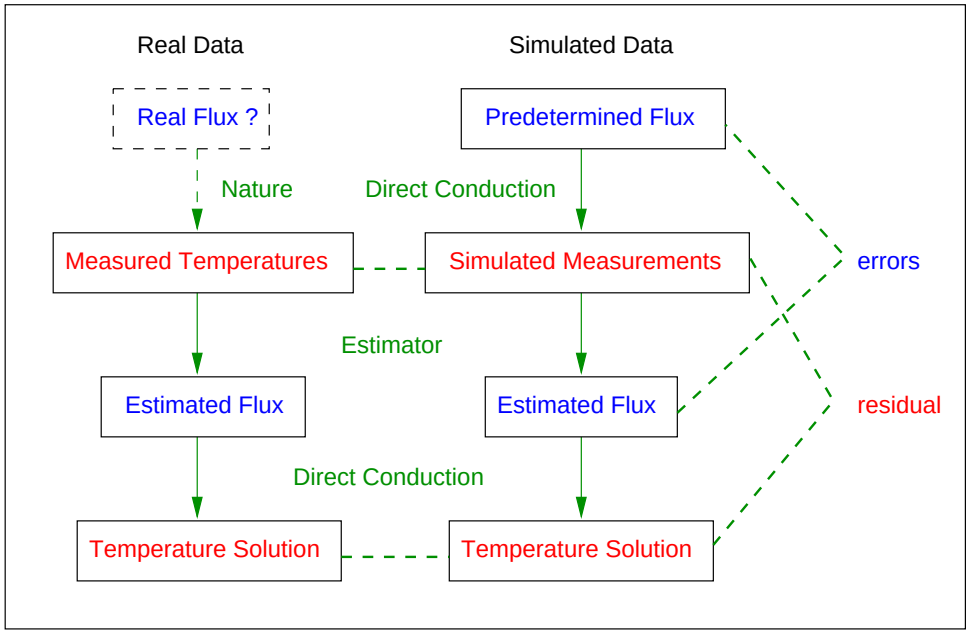

Fig. 1.4 Terminology used to reference different levels of data and the processes by which they are converted.

ture solutions similar to the real data case, we can evaluate the estimated flux directly by calculating the error.

The errors and residuals are simply the differences at corresponding positions and times between the two levels of fluxes or two levels of temperatures, respectively. Furthermore, a global performance assessment can be obtained by averaging the point-by-point residual or error through a root mean squared difference or a mean absolute difference. The performance of the estimator, can be judged by examining all errors and residuals. Once the assessment is complete, a statement of accuracy can be made concerning how well the estimated flux matches the real flux.

Chapter 2

Experimental Instrumentation and Configuration

2.1 Flow Structures

A brief look at the flow structures that are capable of destroying components of aerospace vehicles will not only establish a sense of appreciation for the complexity of the work but also help bracket the experimental designs that are examined here. In other words, we must know what we are trying to measure so that our experiment can be designed to capture the pertinent information.

There are primarily six types of shock interactions characterized by the Mach number and "severity" of the interaction. A severe interaction occurs when the intersecting shocks meet with a large angle (towards perpendicular). In Fig. 2.1, the different types of interactions can be found on a cylindrical model where the type varies depending on the region where the interaction occurs. This paradigm is an adaptation of a similar depiction given by Glass (1989). For this case, Type IV interactions (Fig. 2.2) occur where the incident shock is nearly perpendicular to the bow shock surrounding the cylinder. This region is of particular interest because this is the region where the peak heating will occur.

The heating is a result of a supersonic jet within the subsonic region impinging on the cylinder. As the flow approaches the cylinder supersonically, another shock forms much closer to the surface of the model than the original bow shock. The result of the nearby compression is large localized heating. As the jet turns away from the surface, small

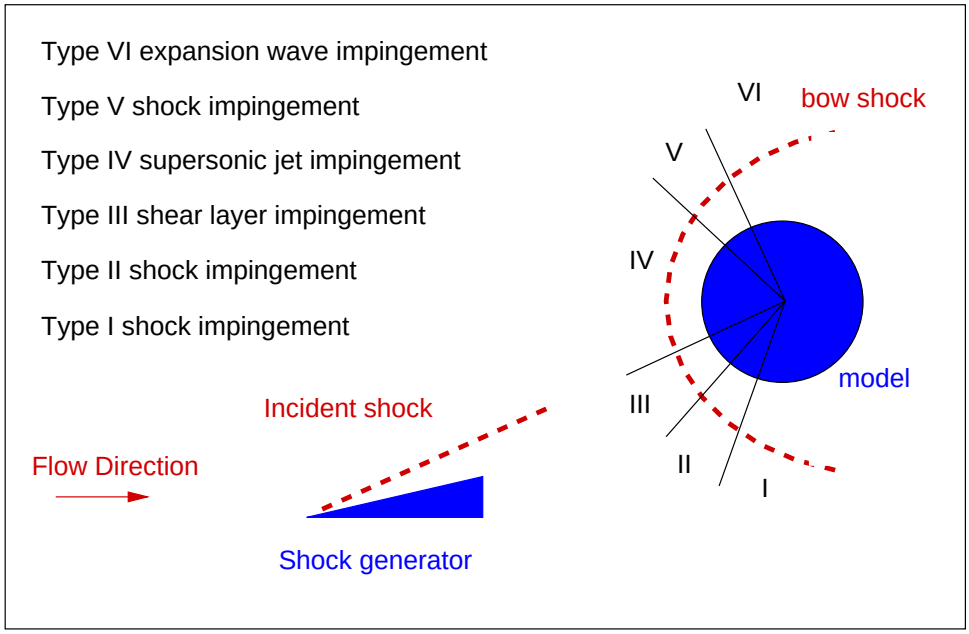

Fig. 2.1 Regions of different types of shock interaction. Type IV shock interactions produce the strongest fluid jets, and therefore represent the regions of highest heating.

expansion and compression waves form along the direction of the flow as shown in Fig. 2.3. These alternating regions of high and low pressure correspond to alternating regions of high and low heating rates as well. (High and low compared to typical stagnation heating.) The largest heating rate would occur in the region directly below the normal shock (region ⑨ in Fig. 2.3 which is an expanded view of the impingement region ⑧ in Fig. 2.2). Away from the impingement on either side, the heating rate will oscillate around the theoretical stagnation heating rate so that underneath the expansion regions we will see a drop in heating of the surface, and under the compression regions we expect a rise in the heating. Furthermore, from Fig. 2.3, larger heating rates would be expected to appear to the right of the impinging shock where the jet layer is smaller. Theoretically this behavior will perpetuate indefinitely. However, in practice, we usually only find one such oscillation apparently due to viscous effects and turbulence which smear the flow structures.

The flows described here produce large, highly localized and sometimes unsteady heating rates. These are the features we want to be able to describe when characterizing the loads potentially placed on aerospace vehicles. To develop materials and designs to

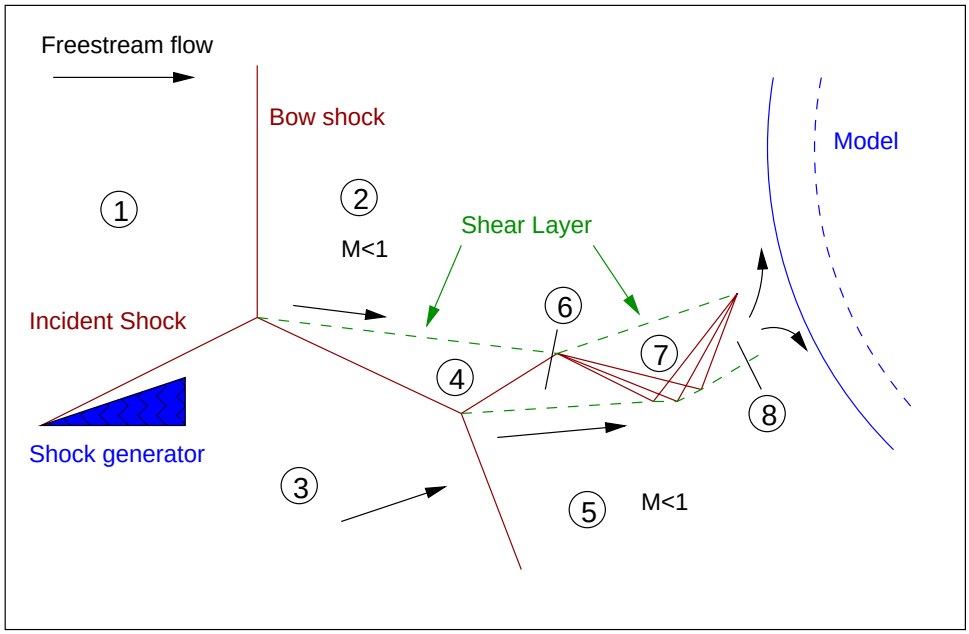

Fig. 2.2 The schematic of the type IV interaction shows the creation of a supersonic jet that impinges the surface of a model. Region ① represents the free stream flow and ③ is the flow behind the incident shock. Regions ② and ⑤ are subsonic behind the bow shock. The supersonic jet separated from the subsonic regions by the shear layer is shown as regions ④, ⑥, ⑦ and ⑧.

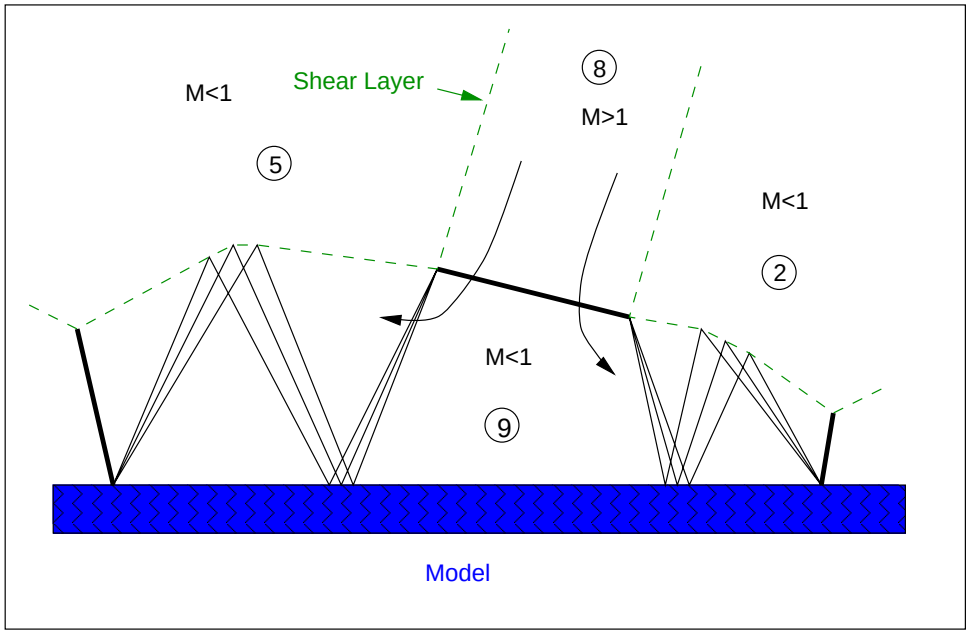

Fig. 2.3 As the supersonic jet (region ⑧) impinges the surface and turns away, regions of alternating compression and expansion develop.

withstand shock interaction heating we must also quantify these loads.

2.2 Experimental Requirements

Even though shock interactions had been studied prior to the X-15 incident, a couple of years passed before it was accepted that shock interactions were the cause of the failure. Upon this realization and since aerospace structures were becoming more complex and faster, experiments designed to study this effect became more common.

The phenomenon of shock interactions are simulated experimentally to obtain a better understanding of the behavior of the nonuniform unsteady heat fluxes created in extreme thermal environments (Hiers and Loubsky, 1967Gladden and Melis, 1994Holden et al., 1988). The concept of the design of the experiment is straight forward enough. We simply generate a planar shock upstream of a test model and configure the tunnel so that the shock intersects the bow shock being shed by the model as suggested by the discussion on flow structures in Section 2.1. Later we will be limiting ourselves to Type IV interactions because of their characteristic large heating rate. Therefore, at the point of interaction a supersonic jet is created which impinges the model.

We want to be able to measure the effects of the impingement, so sensors are placed in the vicinity of the expected impingement. The information we wish to obtain from the experiment is the heating rate of the model due to the flow. However, this typically can not be measured directly (e.g. with a heat flux sensor) because of the spatial resolution required to capture the high speed jet (Miller, 1981). For two-dimensional flows, such as those described in Section 2.1, CFD results (Prabhu, 1994) indicate that the width of the hypersonic jet produced is on the order of 0.6 mm. Assuming a minimum of three sensors are required to resolve the location of the jet, the maximum size of a single sensor is 0.3 mm. Current technology can not provide a heat flux sensor this small. Furthermore, since we want to be able to resolve the relatively small jet as it moves relative to the cylinder, we must place many sensors over a large region. Therefore the cost of heat flux sensors and the lack of resolution, coupled with an environment prone to destroying instrumentation, makes this choice fiscally and practically prohibitive.

Since we are not able to measure the heat flux directly, we are restricted to measuring the temperature and estimating the heat flux (Miller, 1981). Traditionally thin film tem-

perature gauges have been used for this purpose. Recent advances in thin film technology have allowed placement between sensors to be nearly 0.3 mm, achieving the spatial resolution goal. The gauges are either sputtered or hand painted, and are relatively inexpensive. These gauges also have negligible thermal mass (thickness is less than 1 micron, and width is about 0.05 mm), and therefore, have a response time that far exceeds the sample rates used in the experiments. Finally, it should be noted that the gauges usually survive the extreme thermal environments imposed upon them during the experiment.

In an effort to analyze data from experiments, I have chosen two particular configurations which are representative of typical shock-shock interaction experiments. Both experiments described in this work were performed at NASA LaRC by NASA personnel.

2.3 Configuration 1

Based on early experimental work and some analytical work, the canonical shock interaction case study became a cylinder in cross-flow such that the flow structures were two-dimensional in nature. This means that the planar shock will be parallel to the cylindrical model as in Fig. 2.4, and there will be (theoretically) no variation in flow along the length of the cylinder.

The actual flow patterns resulting from the interaction are fairly complex. Aerodynamicists have divided the interactions into six categories based on the location of the interaction of the planar shock on the bow shock (Fig. 2.1). Furthermore, the flows are usually unsteady and nonuniform. Since Type IV interactions produce the largest heating rates (Fig. 2.2), the experiments designed to induce these interactions will be the focus of this configuration. The thin film platinum thermometers are sputtered onto a 7.62 cm diameter disk of Pyrex, which is then inserted into an aluminum cylinder of the same diameter. Ten sensors are spaced radially at 0.8 ° near the expected Type IV interaction region, and another twenty sensors are placed over the remaining test region. The sensors are progressively spaced further apart away from the Type IV region. The temperature sensors are also flanked by pressure transducers to measure the flow variations. This configuration is shown in the insert of Fig. 2.4 (not to scale). Note how there are actually two sections of temperature instruments. In this way the spacing between sensors in the angular direction could be decreased assuming the flow patterns were truly two-dimensional.

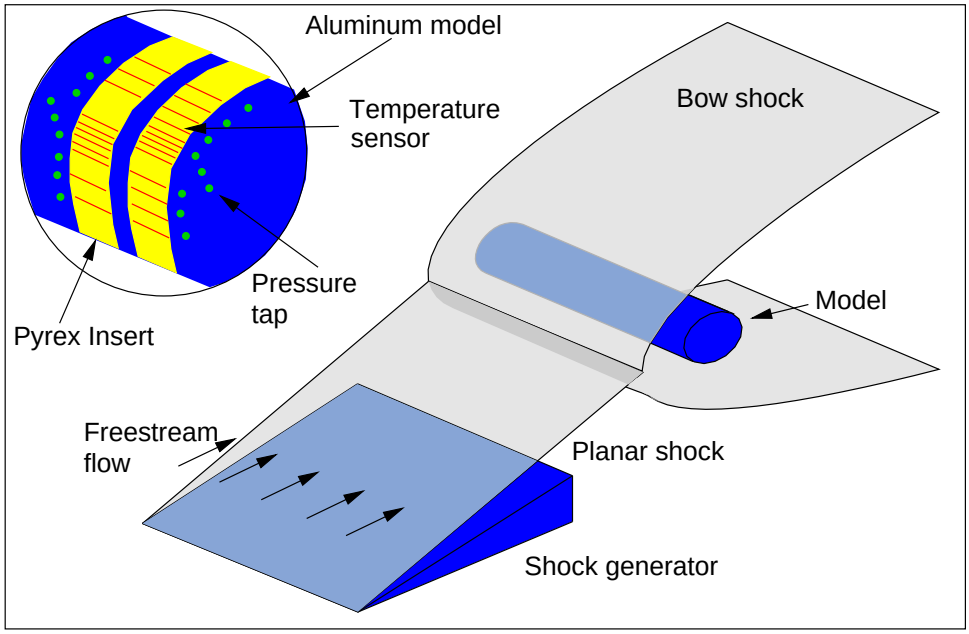

Fig. 2.4 Experimental Configuration 1. The intersection of the shocks is parallel to the cylinder, so the flow pattern is two-dimensional.

The test cylinder and the shock generator are 45 cm wide and are situated such that the shock interaction occurs in region IV, therefore, Type IV jet impingement is present. The tunnel used for this configuration is the *Calspan 48-Inch Hypersonic Shock Tunnel* (48" HST). Our particular test was run using the maximum duration of the facility (20 msec) with a sample rate of 20 kHz for a single sensor (Holden et al., 1988). Therefore, 400 temperature readings were obtained in time for each sensor.

2.4 Configuration 2

A second design for experimentation places a cylinder perpendicular to the incoming planar shock (Fig. 2.5). This configuration was used to study the effects of shock-shock interaction heating on swept wing aircraft, where the cylinder simulated the leading edge of a wing. The flow analysis involved in this case is much more complicated, primarily because the shock interaction is three-dimensional in nature. Therefore, the heating rate will vary along the length of the cylinder as well as around the cylinder.

The cylindrical model, in this case, is only 10 cm long and 1.27 cm in diameter. It was heavily instrumented with tightly packed platinum thin film temperature sensors.

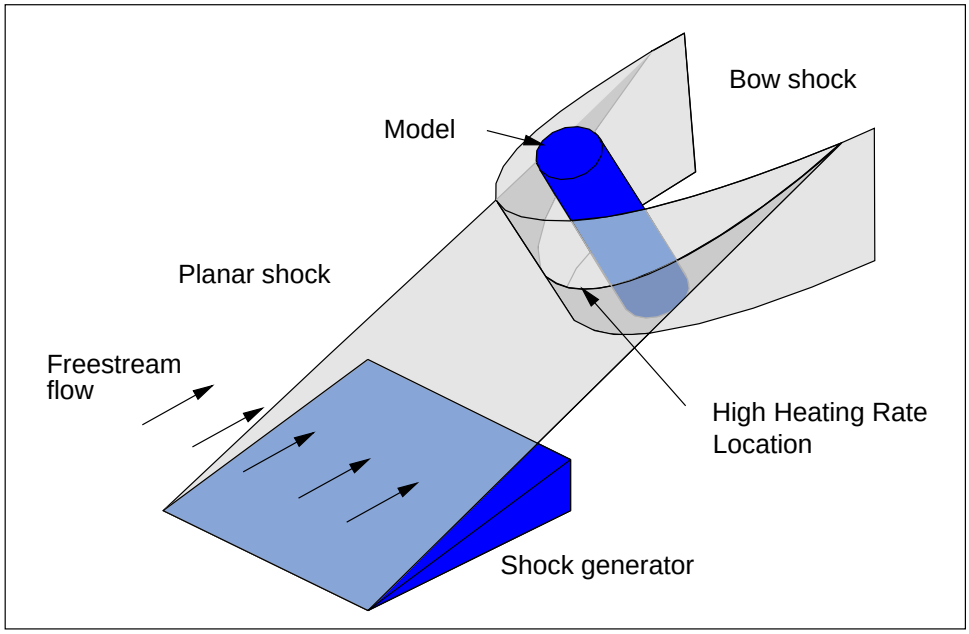

Fig. 2.5 Experimental Configuration 2. The cylinder is perpindicular to the intersection so the flow pattern is three-dimensional.

Two application methods for the sensors were used. The first method was to use vapor deposition of the sensor directly onto the cylindrical Macor model. (Macor is a machinable glass ceramic that is a trademark of the Corning Glass Works). Because the surface of the model is curved, the maximum sensor resolution is 0.635 mm (Miller, 1981). The second method involved depositing the sensors onto a thin (50 μm) flat sheet of Upilex. (Upilex is a polyimide film material and a registered trademark of Ube Industries, Ltd.). The film was then bonded to the Macor core using high temperature epoxy (Duralco #4525). This method allows up to a 0.127 mm sensor spacing. However, the spacing was limited to 0.38 mm due to the leads running to the sensors. The two instrumented models are shown in Fig. 2.6.

Note that these sensors were placed along the cylinder instead of around the cylinder. Similar to the first configuration, however, the sensors were packed as closely as possible in the region of expected maximum heating. Away from the peak region, the spacing was relaxed. There are 70 sensors applied to the bare Macor model and 90 applied to the Upilex/Macor model.

The experiment was run in the NASA Langley 20" Mach 8 Tunnel (Micol, 1995)

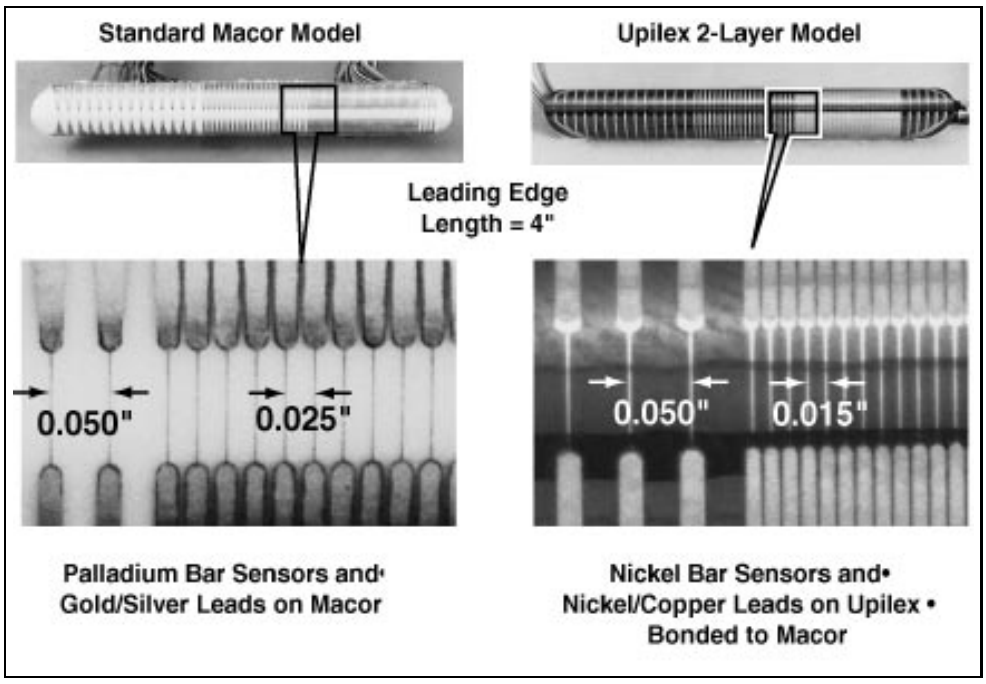

Fig. 2.6 The instrumented models with sensor spacing for both the Macor and Upilex models.

where the duration of the experiment was about 5 sec. Data were recorded every 0.02 sec for all sensors on each model.

2.5 Experimental Data Parameters

These parameters describe the experiments from which the experimental data were obtained. The dimension of the run indicates whether it was used in the unsteady(1) or nonuniform analysis (2). Also note that the NASA LaRC facility is the 20" Mach 8 Tunnel and Calspan facility is the 48" Shock Tunnel. The description of these experiments can be found from Berry and Nowak (1997) for the the NASA data and Holden et al. (1988) for the Calspan data.

Table 2.1 Characteristics of experimental data.

	Run 14 and 36	Run 58 and 60	Run 43	Run 85
Dimension	2	2	1	1
Initial Temperature	300 K	300 K	293 K	301 K
Model	Macor	Upilex/Macor	Pyrex	Pyrex
Sample Rate	50 Hz	50 Hz	20 kHz	20 kHz
Minimum Sensor Spacing	0.635 mm	0.38 mm	n/a	n/a
Duration	4.58 sec	5.12 sec	0.011 sec	0.02 sec
Facility	NASA LaRC	NASA LaRC	Calspan	Calspan

Chapter 3

Traditional Data Reduction

The data reduction problem can be simply solved using one of several different forward techniques. Because temperature measurements are being taken on the surface of the model, the data can be used as a boundary condition to the forward conduction model. Similarly, the boundary flux can be found as a function of the measured temperatures on the surface of a model because the measurements are on the surface. The solution to both approaches is straightforward but the distinction between the type of boundary condition delineates the two classes that comprise the forward techniques. Interestingly, the differences between classes can be described as analytical versus numerical respectively. Briefly, then, Class 1 techniques (*Forward Analytic*) use the flux as a boundary condition and are analytical in nature, while Class 2 techniques (*Forward Numeric*) use the measured temperature as a boundary condition and are numerical.

3.1 Class 1 Techniques

To obtain an analytic solution for the boundary flux we must first relate temperature to an arbitrary flux using the conduction equation. The resulting temperature solution is then a function of time, space and the unknown flux. To obtain the closed-form temperature solution necessary for Class 1 methods, we must require the surface temperature to be piecewise linear, and we must assume that the properties are constant. Use of Laplace transforms (Carslaw and Jaeger, 1959) on the conduction equation results in a surface heating rate as a function of the temperature measurements.

3.1.1 The *Cook-Felderman* Method

An analytical solution, typical of Class 1 methods, was first described by Vidal (1956) and derived by Schultz and Jones (1973). A closed form solution to the data reduction problem is found from the one-dimensional conduction equation. To attain a solution we must first state that we are operating on a semi-infinite domain where the sensor itself is neglected. This assumption is based on the fact that the gauge is small enough so its capacitance does not affect the conduction into the material. We must also require the thermal properties to be constant during the test.

With the surface flux defined as a function of time, $\dot{q}_s(t)$, the conduction problem is given by

$$\frac{\partial^2 T}{\partial x^2} = \frac{1}{\alpha}\frac{\partial T}{\partial t} \tag{3.1a}$$

$$T(t=0) = 0 \tag{3.1b}$$

$$-k\frac{\partial T}{\partial x}\bigg|_{x=0} = \dot{q}_s \tag{3.1c}$$

$$T(x \to \infty) = 0 \tag{3.1d}$$

where the interior is considered insulated. The temperature solution can be found as a function of the boundary flux, time, and the distance into the material. Since our temperature measurements occur on the surface ($x = 0$), the solution, which is written in terms of the Laplace transform (P) is simplified to

$$\overline{T}_s = \frac{1}{\sqrt{\rho c_p k}}\frac{\overline{\dot{q}_s}}{\sqrt{P}}\,. \tag{3.2}$$

Before the transform is inverted, the heat transfer at the surface can be obtained by rearranging Eq. (3.2). The transform and its inversion of the heat transfer are then

$$\overline{\dot{q}_s} = \overline{T}_s\sqrt{P}\sqrt{\rho c_p k} \tag{3.3}$$

$$\dot{q}_s = \frac{\sqrt{\rho c_p k}}{\sqrt{\pi}}\int_0^t \frac{dT(\xi)/d\xi}{(t-\xi)^{1/2}}\,d\xi\,. \tag{3.4}$$

The derivative of the surface temperature in this equation, $dT(\xi)/d\xi$, is the root of the instability of the method. Recall that the data which will be differentiated here is discrete, and any noise will be amplified. Never the less, we can integrate Eq. (3.4) by parts and substitute zero for $T(\xi)$ at $\xi = 0$. This gives the expression

$$\dot{q}_s = \sqrt{\frac{\rho c_p k}{\pi}}\left[\frac{T(t)}{\sqrt{t}} + \frac{1}{2}\int_0^t \frac{T(t)-T(\xi)}{(t-\xi)^{3/2}}\,d\xi\right]. \tag{3.5}$$

Evaluation of the integral in Eq. (3.5) has been discussed by many (Vidal, 1956Henshall and Schultz, 195 but the overwhelmingly most popular approach was described by Cook and Felderman (1966). To continue the solution as described by the *Cook-Felderman* technique, the temperature is assumed to vary linearly between the discrete measurements. After a lengthy manipulation of Eq. (3.5), the surface heating rate is given in terms of the discrete temperature measurements as

$$\dot{q}_n(t) = \sqrt{\frac{\rho c_p k}{\pi}} \left[\sum_{i=1}^{n} \frac{Y(t_i) - Y(t_{i-1})}{(t_n - t_i)^{1/2} + (t_n - t_{i-1})^{1/2}} \right] \tag{3.6}$$

where Y represents the temperature measurement. Note that the properties (ρ, c_p, and k) must be evaluated at some nominal temperature (usually the initial temperature), but can not display temperature dependence.

The fact that the properties can not be temperature dependent, can become significant if we are dealing with large temperature variations such as those found in shock interaction experiments. As a result, Cook (1970b) has since suggested that a correction factor be added to the estimate to account for temperature dependent properties; this approach has been used with some success (Hollis, 1995). The heating rate is corrected for temperature dependent properties by the use of the material property β', which is described by Hollis (1995). The corrected heating rate is then

$$\dot{q}_n = \dot{q}_{on}[1 + \beta'(Y_n - Y_0)] \,, \tag{3.7}$$

where β' is generally a function of Y_n, and now the uncorrected heat rate from Eq. (3.6) is written as the nominal heating rate $\dot{q}_{on}$.

3.1.2 The *Diller* Method

A direct descendant of the *Cook-Felderman* method is the *Diller* method (Diller, 1996). As before, we are operating on a semi-infinite domain with constant thermal properties. Therefore the governing equation is the same as for the *Cook-Felderman* method (Eq. (3.1)). Again using Laplace transforms, the temperature at the surface can be found as Eq. (3.2). Finally, the heat transfer is expressed as Eq. (3.4). The deviation of the *Diller* method from the *Cook-Felderman* method occurs in the representation of the derivative inside the integral.

As explained by Diller (1996), digitally sampled data do not contain any information about the temperature change between measurements. So, while a linear change between two measurements might be reasonable, it would be equally plausible to assume a step change at the beginning or the end of the time period. Fig. 3.1 demonstrates the difference between the *Cook-Felderman* assumption and other plausible assumptions which exhibit different levels of stability. In the *Diller* method, the integral in Eq. (3.4) is evaluated assuming a step in temperature at the beginning of each time period. This method is presumably the most stable solution we could expect from a Class 1 method.

After an evaluation of the integral, the heat rate can be written as

$$\dot{q}_n = \frac{\sqrt{\rho c_p k}}{\sqrt{\pi \Delta t}} \sum_{j=1}^{n} \frac{Y_j - Y_{j-1}}{\sqrt{n-(j-1)}}, \tag{3.8}$$

where we are restricted to a constant time step. This formulation has the distinct drawback of trading accuracy for stability. In other words, the solution would lag due to the increased stability. To correct for this effect, the formulation was refined (Diller and Kidd, 1997) to include additional terms as in

$$\dot{q}_n = \frac{\sqrt{\rho c_p k}}{\sqrt{\pi \Delta t}} \left[\sum_{j=1}^{n} \frac{Y_j - Y_{j-1}}{\sqrt{n-(j-1)}} + (T_{n+1} - T_n) + \frac{1}{3}(T_n - T_{n-1}) \right]. \tag{3.9}$$

The additional information was designed to mimic a control system. The first additional term contains future temperature similar to a feed-forward control function and the second term increases the influence of the current measurement.

3.1.3 The *Kendall-Dixon* Method

Kendall and and (1966) suggested a variation on the derivation of the *Cook-Felderman* method to help smooth the inherent instabilities in the solution. Unlike the *Diller* method, this approach is not simply a different way to evaluate the derivative in Eq. (3.4). In this method, referred to as *Kendall-Dixon*, the solution to an integrated (total) energy is found from the *Cook-Felderman* like expression:

$$q_n(t) = \frac{\sqrt{\rho c_p k}}{\sqrt{\pi}} \sum_{j=1}^{n} \left[\frac{Y_{j-1} + Y_j}{(t_n - t_{j-1})^{1/2} + (t_n - t_j)^{1/2}} \right] \Delta t\,. \tag{3.10}$$

This quantity ($q_n(t)$) representing cumulative energy in the system, is differentiated with a high order central differencing scheme. The heating rate at the n^{th} time step is given by

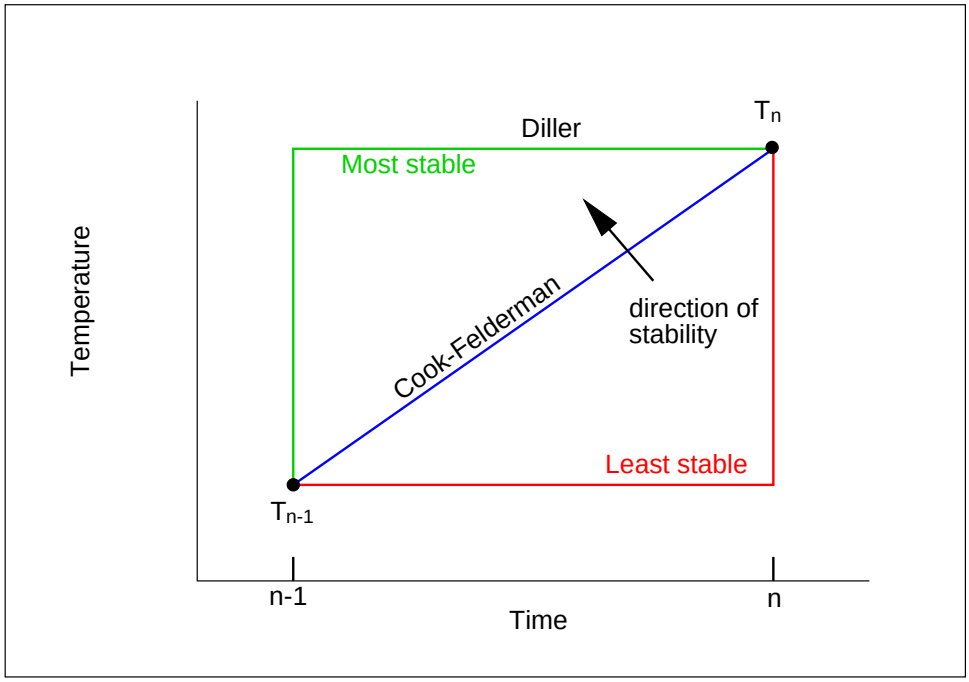

Fig. 3.1 Stability of a Class 1 technique shown as a function of the assumption of the temperature distribution between sensors.

the derivative

$$\dot{q}_n(t) = \frac{dq_n(t)}{dt} = \frac{-2q_{n-8} - q_{n-4} + q_{n+4} + 2q_{n+8}}{40\Delta t}. \tag{3.11}$$

The differencing scheme used here was suggested by Ehrich (1954) and then Hedlund et al. (1980). This approach has been widely used (Hollis, 1995) with the same temperature correction as suggested with the *Cook-Felderman* technique (Eq. (3.7)).

Because of the broad differencing scheme, this method should not exhibit sensitivity to abrupt temperature changes, which should greatly enhance its stability. However, the ability to resolve high frequency components has been traded for the enhanced stability.

3.2 Class 2 Techniques

In response to the primary drawbacks of Class 1 methods (i.e., the restrictions on the functional form of the temperature between measurements and the constant property assumption), the trend has been to use numerical schemes rather than analytical schemes to resolve the temperature distribution. Class 2 methods are therefore characterized by the

use of the measured temperature as the surface boundary condition. The methods in this classification simply require finding a discretized derivative of the temperature distribution at the surface to obtain a flux.

3.2.1 The *Finite-Difference* Method

The most straight forward implementation of the Class 2 methods is the *Finite-Difference* scheme. As with the Class 1 techniques, we are operating on a semi-infinite slab, but now the thermal properties are allowed to change with temperature. As a result, the solution technique will be non-linear, and will probably require some iteration. As in all Class 2 cases, the measured temperature is the surface condition used to calculate the temperature distribution from the conduction governing equation that is given by

$$\frac{\partial}{\partial x}\left[k(T)\frac{\partial T}{\partial x}\right] = \frac{1}{\rho c_p(T)}\frac{\partial T}{\partial t} \tag{3.12a}$$

$$T(t=0) = T_0 \tag{3.12b}$$

$$T(x=0, t=t_n) = Y_n \tag{3.12c}$$

$$T(x\to\infty) = T_0\,. \tag{3.12d}$$

Note that either a finite difference or a finite element solution technique can be used, and they will ideally result in the same temperatures at the given nodes. It is therefore assumed that an appropriate numerical technique is used to resolve the transient temperature response on the interior of the material.

Once the temperature distribution is known, the flux can be found by performing an energy balance at the surface control volume and solving for the surface flux. Normally the surface flux is calculated using Fourier's law. Note that with discrete temperatures in space, the formulation must include the storage term of the surface control volume. Therefore, the formulation for the surface flux on the outermost control volume is

$$q'' = \frac{\Delta x}{2}\rho c_p\frac{\partial T}{\partial t} + \left(-k\frac{\partial T}{\partial x}\right)\,, \tag{3.13}$$

where the first term represents the storage of the control volume and the second term is Fourier's Law at the interior surface of the control volume. This term has not been written in a discrete form because careful consideration must be given to the differencing scheme. The same scheme that was used to find the temperature distribution must be used to discretize Eq. (3.13) as well. Therefore, finite difference and finite element approaches

may formulate this term differently. Cook (1970a) first suggested an explicit formulation, whereas, an implicit formulation is used here because of enhanced stability and accuracy as suggested by Patankar (1980):

$$q''_n = \frac{\Delta x}{2}\rho c_p(Y_n)\frac{Y_n - Y_{n-1}}{\Delta t} - k(\overline{T})\frac{Y_n - T_n^1}{\Delta x}\,. \tag{3.14}$$

Note that Y represents the measured temperature on the surface for the appropriate time step, and that the superscript on the calculated temperature T represents the first nodal value in the interior of the material at the appropriate time. Since the thermal conductivity is temperature dependent, it has been written as a function of some average temperature. Again, it is important to calculate this averaged value as it was calculated in the forward conduction solution method.

Despite the ability to model temperature dependent properties, all Class 2 methods suffer from grid resolution problems. In other words, it will be difficult to reach a point of grid independence when evaluating the derivative of the temperature at the surface. Actually, it will be difficult enough to determine whether grid independence has been reached because the solution converges slowly with decreasing grid spacing. Because we are dealing with such large gradients (large heat fluxes) at the surface, the issue of grid independence becomes even more difficult to verify. Furthermore, the steepness of the gradients increases the sensitivity of the solution to small changes in the measurements.

Note that the *Finite-Difference* method has not been formally described or examined in the literature for this application because it is merely an extension of a numerical conduction problem.

3.2.2 The *Rae-Taubee* Method

Another particularly successful finite difference method that has been well documented is the *Rae-Taubee* technique. First described by Dunn et al. (1986) and summarized by Glass (1989), the conduction equation is transformed into a quasi-linear differential equation using Kirchoff's transformation:

$$\phi = \int_{T_o}^{T} \frac{k(T)}{k_o}\,dT\,. \tag{3.15}$$

Here we are operating on the semi-infinite slab assumption and can incorporate temperature dependent properties. The method is then further refined by scaling the spatial dimension

by the penetration depth to obtain a new coordinate, η, where

$$\eta = x/2\sqrt{\alpha_o t}\,. \tag{3.16}$$

The spatial domain now changes with time and follows the penetration depth of the total added energy. The obvious advantage of this feature is an increase in the resolution of the temperature distribution at early time steps which allows more accurate estimation of the steep slopes and small depths associated with high heat fluxes. This feature reduces the issue of grid independence at early times. However, at later times the formulation is no better than a method using a fixed grid. The conduction equation is now written as

$$\frac{\alpha}{\alpha_o}\frac{\partial^2\phi}{\partial\eta^2} + 2\eta\frac{\partial\phi}{\partial\eta} = 4t\frac{\partial\phi}{\partial t} \tag{3.17a}$$

$$\phi(t=0) = 0 \tag{3.17b}$$

$$\phi(\eta=0, t=t_n) = \int_{Y_o}^{Y_n} dT \tag{3.17c}$$

$$\phi(\eta=5) = 0 \tag{3.17d}$$

where the formulation is in terms of the Kirchoff temperature defined in Eq. (3.15). To maintain a semi-infinite slab assumption, the temperature at five penetration depths ($\eta = 5$) is zero. The reference conditions (indicated by the naught subscript) are evaluated at the initial temperature. Any appropriate finite difference or finite element solution method can be used to find the scaled temperature as a function of space and time.

To obtain the estimated heat flux, the temperature distribution is differentiated at the surface. If a Crank-Nicholson differencing scheme is used, the heating rate becomes

$$\dot{q}_n = -k_o\frac{1}{2\sqrt{\alpha_o t_n}}\left[\frac{-3\phi_n^0 + 4\phi_n^2 - \phi_n^2}{2\Delta\eta}\right]. \tag{3.18}$$

The temperature on the surface ϕ_n^0 is found using the Kirchoff transformation of the measured temperature. In this formulation, the capacitance of the surface control volume has been neglected. Again, for early times, this may be reasonable because the control volume is so small. However, at later times, this will bias the solution.

As pointed out by Holden and Kolly (1995), the scheme (*Rae-Taubee* method) is convenient because regridding is automatic. George et al. (1987) performed an evaluation of this technique compared to previously known methods. They found that the *Rae-Taubee* technique could resolve early heat fluxes as well or better than other schemes. Note, however, that the *Rae-Taubee* technique suffers from inadequate grid spacing at later times as previously suggested.

3.2.3 The *Simple-Implicit* Method

Holden and Kolly (1995) reported that many techniques are not able to resolve high frequency components of the estimated flux (speaking primarily of the *Rae-Taubee* technique). This drawback gave rise to what is called the *Simple-Implicit* technique. It has been found that the conduction equation cast into its Kirchoff form and discretized implicitly is able to resolve the flux at early times (with appropriate gridding) while adequately estimating high frequency components at later times. For this method we begin with the Kirchoff formulation of the conduction equation:

$$\alpha(\phi)\frac{\partial^2\phi}{\partial x^2} = \frac{\partial\phi}{\partial t} \tag{3.19a}$$

$$\phi(t=0) = 0 \tag{3.19b}$$

$$\left.\frac{\partial\phi}{\partial x}\right|_{x\to\infty} = 0 \tag{3.19c}$$

$$\phi_n(x=0) = \int_{T_o}^{T_n} \frac{k(\xi)}{k_o}\, d\xi\,. \tag{3.19d}$$

Again the reference value is taken as the initial measurement.

The difference formulation is given as

$$\frac{\phi_i^{n+1} - \phi_i^n}{t_{n+1} - t_n} = \alpha(x_i, t_n)\frac{\phi_{i+1}^{n+1} - 2\phi_i^{n+1} + \phi_{i-1}^{n+1}}{\Delta x^2} \tag{3.20}$$

where the Kirchoff temperature is given as in Eq. (3.15). This method allows for relatively simple implementation of nonuniform gridding schemes and linearizes non-linear problems. Also since we now are concerned with temperature differences (ϕ) rather than the actual temperatures, the transformation reduces the impact of steep gradients for both linear and non-linear cases. However, the surface derivative is still sensitive to measurement errors. Furthermore, the standard method of flux estimation from the distribution is to simply take the derivative at the surface. This route however, ignores the energy stored in the surface control volume as described earlier in Eq. (3.13).

3.3 Summary of Class 1 and Class 2 Methods

Traditional data reduction techniques have been divided into two classes based on their solution approach. Class 1 methods are analytically based, and suffer in general from modeling constraints. Since the solution is closed form, variable temperature properties

and complex conduction models (anything other than a single material, semi-infinite slab) can not be considered. Being able to characterize the test model accurately will be shown to become significant for shock interaction data. Class 2 methods, which are numerical, can include complex modeling situations but are limited by gridding concerns. Even when the temperature distribution can be found with a high degree of confidence, the gradient at the surface usually can not be resolved with the same accuracy as the temperature.

The progression of the development of the techniques provides a unique insight into each methods' drawbacks and the proposed solution to that drawback. For example, the *Cook-Felderman* method, which was the first important development in heat flux estimation, was found to be sensitive to measurement errors. The *Kendall-Dixon* method was an attempt, then, to dampen the amplifications inherent in this method. Similarly, the *Diller* method addresses this concern with somewhat improved results. Because of the shortcomings of Class 1 methods, due to their analytic nature, Class 2 methods were designed to generate a solution method that could incorporate thermal properties that are dependent on temperature. The numerical temperature distribution (given the measurement as a boundary condition) is differentiated at the surface to obtain a heat flux. The *Finite-Difference* method, as an example, does this, but it has serious gridding problems. Therefore, the *Rae-Taubee* technique was conceived to mitigate the gridding problem. Despite its apparent usefulness, it was found to have difficulties at later times. As a compromise, the *Simple-Implicit* scheme was formulated and has been considered the most adequate scheme, until now.

Both classes involve calculating a differential at the surface to obtain the flux. For Class 1 methods, the derivative is performed analytically, but the evaluation uses discrete data. Likewise, Class 2 methods must approximate the derivative with a finite difference formulation. In both cases, the differentiation of discrete data results in instabilities. These drawbacks have lead to the requisition of the Class 3 techniques for greater accuracy and reliability.

Chapter 4

Inverse Solution Techniques

The third class of methods, characterized by inverse heat conduction procedures, is new to this application (Walker and Scott, 1995). The *i*nverse *h*eat *c*onduction *p*roblem (IHCP) is identified as the boundary identification problem from interior temperature measurements (Sparrow et al., 1964). Although the problem under consideration is not a true inverse problem because the temperature measurements are on the surface of the model, an inverse technique based on a least squares minimization is being proposed due to the high uncertainties in the measured temperatures and inherent instabilities in the problem.

The IHCP and its solution was successfully characterized along with potentially the first general solution by Stolz, Jr. (1960). An integral transform technique was used to solve the integral heat conduction problem similar to that given in Eq. (1.1). Soon afterwards, Beck (1968) fathered the approach that is now considered the base or standard from which most other techniques are compared. His method involved a least squares minimization and was found to be more flexible than Stolz's method.

The inherent non-linearity of our problem (due to temperature dependent properties) lends itself well to numerical solution techniques. Beck (1970) introduces what is now called "Beck's second method" to handle non-linear problems. Using this technique, Bass (1980) demonstrates the use of finite element routines in solving IHCP's. Interestingly, Bass used a periodic flux to evaluate his solution method as will be done here. However, the frequency was much lower than that of interest in this work. These pioneering inverse triumphs provide the basis for the inverse solution techniques examined here.

The basic approach of the inverse technique can be seen in Fig. 4.1. For our purposes,

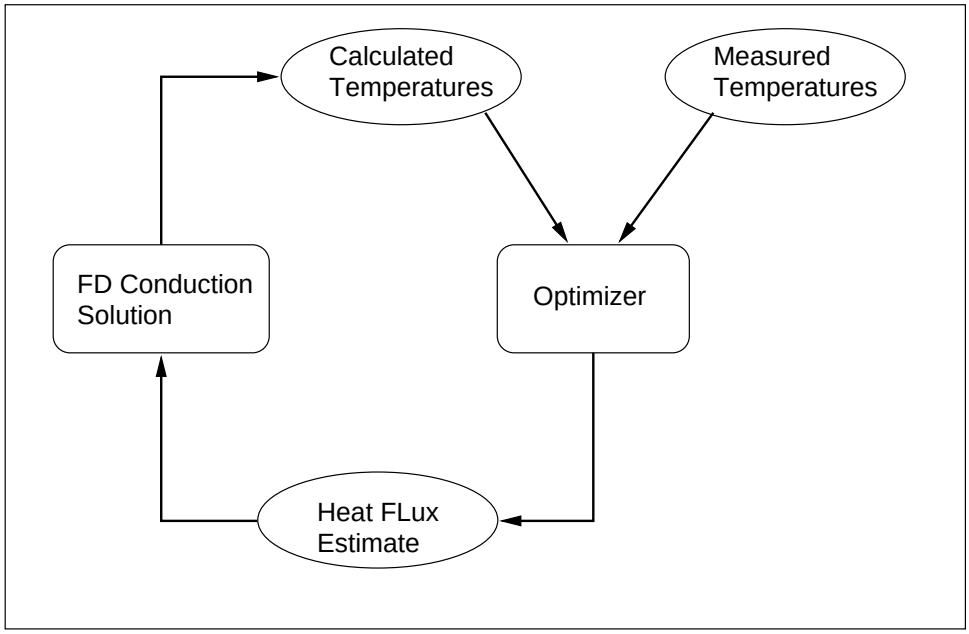

Fig. 4.1 The inverse solution approach

the optimization block in the diagram involves a least squares minimization routine. An algorithm for the inverse solution along with a more detailed explanation of the optimization is shown in Fig. 4.2 in the form of a flowchart. Each piece of the algorithm will be described in greater detail when each individual implementation is presented.

The solution (heat flux estimate) is found by first calculating surface temperatures from an initially assumed heat flux. This forward temperature solution can be found by any appropriate conduction solution method such as finite element which can include temperature dependent properties. The calculated temperatures from the forward conduction analysis are then compared to the measured temperatures in a least squares sense. An appropriate optimization scheme can then be used to find an updated guess by minimizing an objective function which involves the difference between calculated and measured temperatures. This procedure avoids differentiating the distribution at the surface, and therefore, reduces the amplification of uncertainty inherent in direct solution techniques (i.e., Class 1 and Class 2 methods).

Several advantages of inverse technology over traditional estimation methods have been suggested at this point. Table 4.1 provides an enumeration of the features of inverse solution methods and demonstrates their ability to estimate accurate solutions with controlled

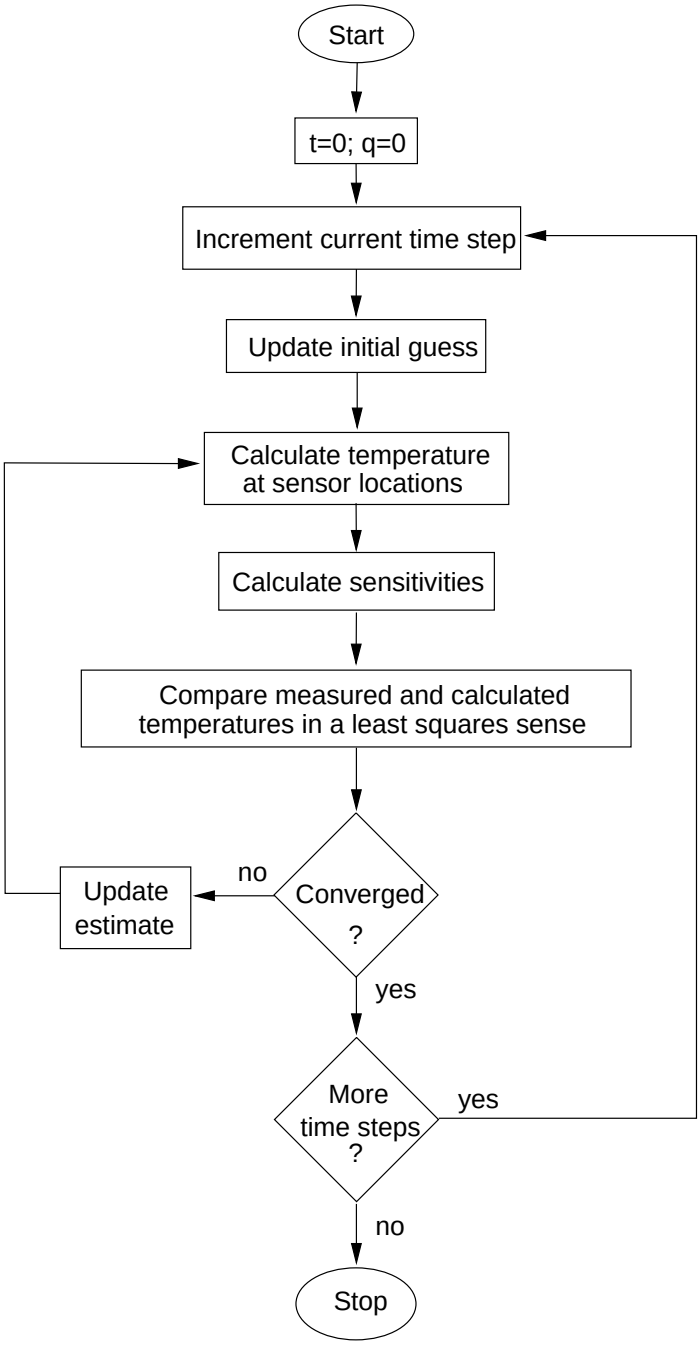

Fig. 4.2 The inverse solution algorithm in the form of a flowchart.

instabilities. The initial advantage and possibly the most pertinent to this discussion is the ability of the inverse method to accurately characterize lateral conduction effects. Because we can easily incorporate a multi-dimensional conduction solution into the inverse algorithm (Beck et al., 1985), the solutions should (and will) be able to describe the heating rates better than one-dimensional techniques. Using multi-dimensional inverse approaches is not new (Imber, 1974), but their application to this problem and specific formulation are. Previous work on multi-dimensional inverse problems typically involve the determination of opposite boundaries from interior measurements (Guo and Murio, 1991Baumeister and Reinhardt, 1987). This and the remaining features will be discussed in detail in the context of the development of the different flavors of inverse techniques. Furthermore, the effects of inverse methods which possess these qualities will be shown through a comparison with the estimates from other methods.

4.1 The *Function-Specification* Methods

The first Class 3 methodology examined is called the *Function-Specification* method. Note that this technique has been applied to this problem previously, as a part of this work, using several test cases with simulated data (Walker and Scott, 1997). In this method, a value for the heat flux is predicted provided the estimate fits a predetermined functional form. We are, in essence, estimating the coefficients of a function that describe the boundary flux. The one-dimensional solution procedure will be described followed by extension on the method to incorporate two-dimensional conduction.

4.1.1 One-Dimensional Approach

For the one-dimensional approach, we will be estimating a heat flux as a function of time. Inverse procedures can produce a solution based on a whole domain approach or a sequential approach. Note that our estimates are found in a sequential manner, such that only the current time and a "few" future times are examined simultaneously, because of the efficiency of sequential estimation. The motivation for a sequential approach is founded in the fact that heat travels in a solid diffusively. This means that an interior response to a change in a boundary condition lags and damps the original signal. This means we must look forward in time to determine the response. Furthermore, long term effects are virtually undetectable

Table 4.1 Inherent advantages of the inverse methods over traditional methods.

Advantages
1. Extendibility to multiple dimensions
2. Inclusion of non-constant variance of measurement noise
3. Lack of derivative approximations at the surface
4. Lack of grid independence issues
5. Ability to model complex physical systems

compared to short term effects. This means that we should limit our forward looking to "nearby" times. What is meant by "nearby" must be determined for each problem.

The type of function must be determined as well as the number of simultaneous time steps. Usually we assume the function is either a constant or a line, but it can theoretically be anything. If we select a constant, we are assuming that the flux will be constant over the given number of time steps. For a given value of the constant heat flux we can calculate a temperature response from a forward conduction solution. Now we want the difference between the calculated temperatures and the measured temperatures to be at a minimum. This can be accomplished by minimizing the sum of the squares of the difference. Once the value of the constant flux is found, all but the current time step is discarded. The method then progresses to the next time step where the procedure is repeated. Note that the last few times can not be found with any confidence because the methodology requires a certain number of future times.

The optimal number of future times R must be determined for each problem. Factors that influence the selection of R are the measurement times, the relative amount of noise in the measurement signal and possibly other thermal characteristics of the problem. Never the less, a lower bound on the number of times can be determined. To achieve any biasing, a *Constant-Specification* method must examine at least two simultaneous times ($R \geq 1$). Similarly, a *Linear-Specification* method must examine at least three simultaneous times, etc. If we restrict a *Constant-Specification* method to a single time step, the method reduces

to what is called an *Exact-Matching* technique, where there is no bias added to dampen the instabilities.

In our formulation of the *Function-Specification* methods, the appropriate number of future times will be the lower bound for any order function chosen. This can be seen once we remember the reason for future times. There is a delay from the time the heat flux is applied to the time when the interior sensor responds, hence the need to look forward. However, our sensors are on the surface and the response is theoretically instantaneous. Therefore, the minimum number of future times is required.

The formulation for the function coefficients that make up the estimates is found from minimizing the objective function which is the sum of squares of the difference between measured ($\vec{Y}$) and calculated ($\vec{T}$) temperatures, given by

$$S = [\vec{Y} - \vec{T}]^{\mathsf{T}}[\vec{Y} - \vec{T}]\,. \tag{4.1}$$

Here, the size of the vectors is $R+1$. To solve for the heat flux, we first make an assumption as to the functionality of the flux. Common choices (and ones that will be implemented here) are constant and linear. Next the calculated temperature vector is written as a Taylor series expansion and contains the temperatures that are being considered:

$$\vec{T} = \vec{T}_o + \mathbf{X}\vec{q}. \tag{4.2}$$

If the current flux estimate is given by q_o then the temperature response to that flux is T_o and the sensitivity $\mathbf{X}$ is defined as the derivative of the temperature with respect to a change in flux. Note that this formulation would be exact for a linear conduction problem. The sensitivity matrix, $\mathbf{X}$, represents the change in temperature due to a change in heat flux, where the components are given by

$$X_{ij} = \left.\frac{\partial T_i}{\partial q_j}\right|_{\vec{q}_o}\,. \tag{4.3}$$

The elements are calculated by finding a new temperature response to a perturbed heat flux. Note that the elements above the diagonal will be zero because previous temperatures are not affected by changes in later fluxes. Note that the calculated temperatures are found from any legitimate conduction solution.

The objective function must now be minimized. To minimize Eq. (4.1) (the objective function), the derivative with respect to the flux is set equal to zero. The matrix algebra

leaves us with a linear set of equations given by

$$\mathbf{X}^{\mathsf{T}}\mathbf{X}\vec{q} = \mathbf{X}^{\mathsf{T}}(\vec{Y} - \vec{T}_o)\,. \tag{4.4}$$

This can easily be solved to obtain a new flux estimate $\vec{q}$. Note that for nonlinear conduction, the ultimate estimate for each time must be found iteratively. For each iterative step the sensitivity matrix must be re-evaluated until the change between successive iterations is adequately small.

4.1.2 Two-Dimensional Approach

At this point we have described the typical one-dimensional *Function-Specification* solution technique. However, the goal is to enhance the solution methodology to include multi-dimensional conduction. Therefore, we must describe the implementation of the *Function-Specification* method in the spatial direction. Again, a functional form of the boundary flux (which is now a function of location as well as time) must be predetermined, and the coefficients that minimize the objective function must be found. Note that there is no requirement concerning the order of the spatial function relative to the temporal function. For example, we are free to select a constant function in time and linear function in space. The determination must be made based on the parameters in the problem.

One issue that becomes confusing concerning two-dimensional estimation is the computational treatment of the spatial direction. For efficiency in the one-dimensional method, we were able to solve for the successive time steps in a sequential manner. Since the spatial direction is elliptical, rather than parabolic in nature, the inclusion of neighboring sensors in the estimation procedure will help introduce a bias into the solution much like the future time steps did for the temporal direction. However, the solution can not march in the spatial direction as it did in the temporal direction. As a result, all spatial estimates for a given time must be found simultaneously.

The other issue that complicates the formulation is the treatment of spatial and temporal data in the vectors. For the one-dimensional method, we constructed vectors where each element represents a different time. Now we must incorporate spatial information as well as temporal information. Here, the components of the temperature vectors consist of temperatures from different sensors at a single time followed by the temperature for the same sensors at the next time step. For example, if we use subscripts to denote spatial location

and superscripts to denote time step, the temperature vectors ($\vec{T}$ and $\vec{Y}$), considering two time steps and two spatial locations, can be written as

$$\vec{T} = \{T_1^1, \quad T_2^1, \quad T_1^2, \quad T_2^2\}^\mathsf{T} . \tag{4.5}$$

The sensitivity matrix must also contain spatial information. The construction of this matrix then, must be consistent with the construction of the temperature vectors. As a result, the sensitivity matrix should be written as

$$\mathbf{X} = \begin{bmatrix} X_{1,1}^{1,1} & X_{1,2}^{1,1} & X_{1,1}^{1,2} & X_{1,2}^{1,2} \\ X_{1,1}^{2,1} & X_{1,2}^{2,1} & X_{1,1}^{2,2} & X_{1,2}^{2,2} \\ X_{2,1}^{1,1} & X_{2,2}^{1,1} & X_{2,1}^{1,2} & X_{2,2}^{1,2} \\ X_{2,1}^{2,1} & X_{2,2}^{2,1} & X_{2,1}^{2,2} & X_{2,2}^{2,2} \end{bmatrix} \tag{4.6}$$

where the components are defined as

$$X_{i,j}^{m,n} = \frac{\partial T_i^m}{\partial q_j^n} . \tag{4.7}$$

The values of the sensitivities defined in Eq. (4.7), can be found several ways. The approach used here is by no means the ideal solution. Appendix A briefly explains a more rigorous treatment of the sensitivity calculation. The straightforward perturbation method was used here due to its simplicity and straightforward application. If the current estimate is perturbed, a new temperature signature will be obtained. The difference in the temperature responses divided by the heat flux perturbation is the sensitivity we are seeking. More explicitly, the expression for the sensitivity calculation is given as

$$X_{i,j}^{m,n} = \frac{T_i^m(\vec{q_j^n} + \delta q) - T_i^m(q_j^n)}{\delta q} , \tag{4.8}$$

where δq represents the perturbation over an area. Despite the apparent simplicity, the perturbation that is performed must be chosen with care. As described in a previous paper (Walker and Scott, 1997), the sensitivity "patch" (perturbation) determines, in some respect, the amount of bias that is introduced into the solution. In other words, where and how the current estimate is perturbed, can affect the stability and accuracy of the solution significantly.

Two different types of patches were used for the analysis of nonuniform heat fluxes. These two, are both actually constant specification methods (as discussed in Section 4.1) in

space whereas we normally apply this method in time. It is immediately apparent that the constant patch is a constant specification method. We are estimating a single value for the flux estimate over a region as seen in Fig. 4.3. The triangular patch then would appear to be a linear specification. Notice however, that we are estimating a single value (that being the peak) for this patch. The ends of the triangle are fixed at zero. Notice how this patch redistributes the zone of influence; the perturbation is felt over more sensors. Also realize that the addition of the patches for all sensors results in a consistent perturbation over the entire domain.

Note that to increase the bias, the span of the patch can be stretched to include two sensors on each side. This is equivalent to increasing the number of future times. In this case, a linear distribution from one (at the center) to zero (two sensors away from the center) would constitute the perturbation. Note that the estimate obtained for the sensor in question (the sensor in the center) must be distributed over the patch region in accordance with the weighting dictated by the patch. In other words, we must multiply the flux correction for a particular sensor by the same patch that was used to calculate the sensitivities.

The freedom to construct different patches is limited only by the requirement that the addition of all the patches adds to a constant over the domain. Fig. 4.4 demonstrates this concept for clarity. This result is that we can extend the patch as far as necessary to obtain a stable solution. In fact, the types of patches can be mixed as shown in Fig. 4.4 for situations where the sensitivity changes due to sensor spacing, etc. Despite the potential time savings of the alternate sensitivity calculation method given in Appendix A, this direct approach provides additional flexibility.

The techniques used in the two-dimensional *Function-Specification* solution method employ the triangular patch that spans two elements as shown in Fig. 4.3 for each sensor in the domain. This configuration will be shown to provide satisfactory estimates without further manipulation of the type of patch for different sensor configurations.

4.2 The *Regularization* Method

Where the *Function-Specification* methods introduce bias by fixing the functional form of the estimate over a limited time span, the *Regularization* method simply adds a term onto

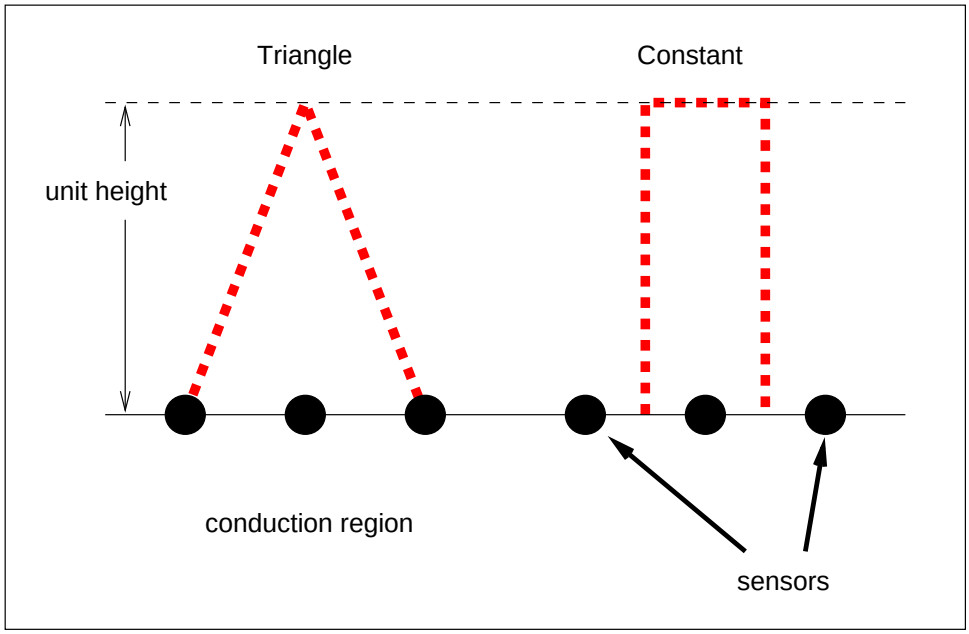

Fig. 4.3 The triangular and constant patch used for two-dimensional inverse estimation calculations.

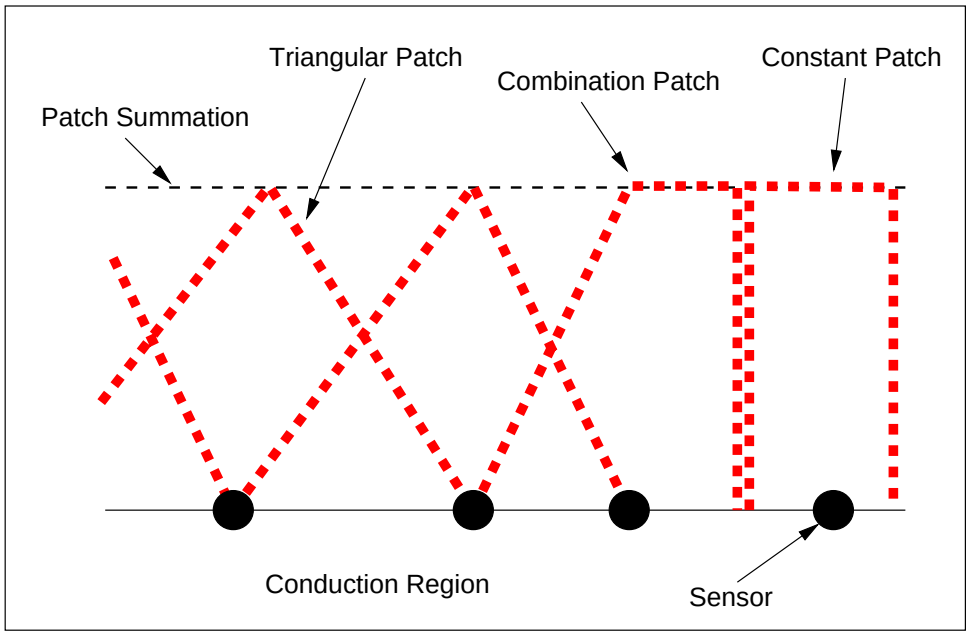

Fig. 4.4 Demonstration of the summation of different styles of patches.

what is referred to as the objective function given for the *Function-Specification* method as Eq. (4.1). This embodies the primary difference between the formulation and solution methodology of the two methods. This type of bias is usually less restrictive, because the function of the estimate can be anything. The biasing occurs because the additional term has the tendency to force the shape of the relationship of the estimates towards a particular order of function. Note that this approach will also be sequentially as was the *Function-Specification* method.

The objective function for the second of the Class 3 techniques is formulated by introducing a bias term (regularization term) to a sum of squares of the temperature differences and is given by

$$S = [\vec{Y} - \vec{T}(\vec{q})]^{\mathsf{T}} \Psi^{-1} [\vec{Y} - \vec{T}(\vec{q})] + \alpha (\mathbf{H}\vec{q})^{\mathsf{T}} (\mathbf{H}\vec{q}) \tag{4.9}$$

where the regularization term is $\alpha(\mathbf{H}\vec{q})^{\mathsf{T}}(\mathbf{H}\vec{q})$. This formulation is typical for true inverse problems and is referred to as the *Regularization* technique. The formulation follows that of Tikhonov and Arsenin (1977) except that the matrix of measurement variances (Ψ^{-1}) has been included as described by Beck (1970). Through this matrix, non-constant variance errors can be included for situations when the noise can be characterized and is variable.

In our case, first order regularization is used ($\mathbf{H} = \mathbf{H}_1$) which limits the flux gradients and tends to force the estimate to a line. That is, $\mathbf{H}_1$ contains the coefficients of a first-order finite difference scheme as shown by Beck et al. (1985). To gain insight into the operation of the biasing term, consider the example of first order regularization. This type forces the estimates toward a constant. This is accomplished by including the difference between sequential estimates. When the objective function is minimized, the global solution will be some combination of the minimized residual (difference between temperatures) and the minimized flux differences. This balance act can be adjusted through the use of the regularization parameter α. This will be discussed in detail later.

To solve for the estimate, the calculated temperatures, $\vec{T}(\vec{q})$, in Eq. (4.9) can be obtained by any appropriate forward conduction solution. As a result we can incorporate temperature dependent properties which cause the solution to become non-linear. Because of this non-linearity, the solution ($\vec{q} = \vec{q_o} + \Delta\vec{q}$, where $\vec{q_o}$ is the flux estimate at the previous iteration) is formulated in terms of a flux correction $\Delta\vec{q}$. The solution procedure then follows that given by the *Function-Specification* method in Section 4.1.1. Using the Taylor

series expansion of the temperature (Eq. (4.2)) and the definition of the sensitivity matrix (Eq. (4.3)), a linear set of equations can be obtained by equating to zero the differentiated objective function. Thus, $\Delta\vec{q}$ can be found from

$$\Delta\vec{q} = \left[\mathbf{X}^{\mathsf{T}}\Psi^{-1}\mathbf{X} + \alpha\mathbf{H}^{\mathsf{T}}\mathbf{H}\right]^{-1}\left[\mathbf{X}^{\mathsf{T}}\Psi^{-1}(\vec{Y} - \vec{T}_o) - \alpha\mathbf{H}^{\mathsf{T}}\mathbf{H}\vec{q_o}\right] . \tag{4.10}$$

The flux estimate is found iteratively by calculating the flux correction based on an initial guess and updating the guess until the correction is arbitrarily small.

Note that we must examine several time steps simultaneously to achieve any biasing as in the *Function-Specification* method. A sequential approach where a minimum number of future time steps is examined at once is the most efficient. Generally, for a sequential solution approach (which was used here), the minimum number of required time steps depends on the order of regularization. In our case, which uses first order regularization suggested by Scott and Beck (1987), the minimum required number of time steps is two. Note that more than two steps introduced too much bias into our problem. This finding is concurrent with those of the *Function-Specification* method.

Another issue affecting the amount of added bias is the magnitude of the regularization parameter α. A rough estimate of the regularization parameter can be determined based on the variance of the error in the measured data as $\alpha = (\sigma_y/q_n)^2$, where the variance given by σ_y is on the order of degrees, and q_n is some characteristic heat flux (often times it is the maximum flux but is problem dependent). This value can be found a priori when a ball park value for the heat flux is assumed and for test cases when the flux is known. The optimal regularization parameter can also be found through trial and error by requiring the objective function value S, to be on the same order of magnitude as the experimental noise. Another time consuming but effective method for determining the regularization parameter, is locating the value of α the gives the lowest error for estimates of test cases. A value of $\alpha = 0.00001\,\mathrm{K^2m^4/W^2}$ was found to satisfy these approaches to determining the regularization parameter; the results of this procedure are presented and explained in Section 6.2.

4.3 The *Explicit-Regularization* Method

Recall that to introduce bias using the *Function-Specification* and the *Regularization* methods, future times must be examined. Also recall that our situation does not fall under the

classification of a true inverse problem because we have surface temperature measurements. As a result, the temperature response to the boundary flux is theoretically instantaneous. Therefore, contrary to a formal inverse formulation, we realize that future time steps are unnecessary. We therefore, wish to introduce a modified regularization procedure called the *Explicit-Regularization* method that does not utilize measurements at future times. The issue that arises then concerns the introduction of bias. If we eliminate the future times, we also eliminate the bias. The new technique will apply regularization to the difference between the current estimate and the *previous* estimate. Since the previous estimate is already known, the current estimate can be found explicitly (hence the name).

The objective function given by Eq. (4.9) will be written in terms of the current time step and the previous estimate which results in a scalar equation. Thus, with first order regularization, the modified objective function becomes

$$S_i = \psi_i^{-1}(Y_i - T_i)^2 + \alpha(q_i - q_{i-1})^2 \tag{4.11}$$

where q_{i-1} is the previous estimate. Note that the variance of the measurement at the current time step is included as in the *Regularization* and can be thought of as a weighting factor. Normally, this value is constant for all time steps. However, it may include unusually large variances due to large measurement errors to be described in Section 5.2.4.

Now, we can explicitly solve for the flux correction as we did for Eq. (4.10) to obtain

$$\Delta q_i = \frac{\psi_i^{-1} X_i (Y_i - T_{oi}) - \alpha(q_{oi} - q_{i-1})}{\psi_i^{-1} X_i^2 - \alpha} \tag{4.12}$$

The regularization was included for completeness, but can be eliminated by setting the regularization parameter to zero. We are left then with a technique which is equivalent to an exact matching technique. Notice that the *Explicit-Regularization* technique is also extremely fast because we can solve for the estimate at each time explicitly without having to invert a matrix as in the *Regularization* technique.

4.4 Summary of Inverse Methods

Despite the fact that our problem does not fit into the realm of a true inverse problem, we have suggested that inverse techniques can be useful in easing the effects of uncertainty in data. On the other hand, the fact that the measurements are on the surface where the

heat flux is to be estimated, the standard inverse routines can be simplified somewhat by considering a single time step instead of several future times, thus the incarnation of the *Explicit-Regularization* method. Inverse technology also offers several features not found in the other classes of solution methods that we can exploit in our new method. For example, since we are not taking derivatives of the temperature distribution, small errors tend not to become amplified in the solution as in the other methods. Since the method is statistical in nature, we can mitigate the influence of noisy data. Also, we can include information about non-constant variance of the measurement errors in the Ψ^{-1} matrix. Finally, multi-dimensional problems can be considered as mentioned previously. This feature is perhaps the most crucial when considering the focus of this work. Ultimately, we would like to resolve lateral conduction effects which will require a true two dimensional estimation routine. It was demonstrated how inverse procedures can incorporate such non-uniformities.

Chapter 5

Performance Evaluation

5.1 Evaluation Tools

The focus of this work is to develop a new method to solve the heat flux from surface temperature measurement data reduction problem, to verify its accuracy and validity and to compare the estimates of the new method to other known standard methodologies. To accomplish the evaluation of the methods, a reliable test procedure must be employed to provide meaningful comparisons. This test procedure encompasses three primary formats:

- Visual inspection of the estimates
- Visual inspection of the residuals
- Quantification of the errors and residuals

The estimates and their residuals will be compared to known temperature histories that were calculated from predetermined fluxes. The set of predetermined fluxes will comprise the data in our test cases.

The qualification as well as the quantification of the estimates and their residuals will facilitate the characterization of the behavior of each method. By examining the estimates compared to each other and compared to a standard, we will be able to determine relative accuracy and leading or lagging effects. We should also be able to induce the type of flaw in the method that causes errant behavior. The residuals can provide information about the stability and accuracy of the solution. These values are found by assuming the estimate to be the boundary condition to a forward conduction problem. For evaluation purposes, this

solution should be independent of the estimation method, and provide the most complete conduction model possible.

Errors in the estimate and residuals can be quantified by finding a representative average of the deviation. In this case, two methods have been employed:

$$\text{Root Mean Square Error} \qquad e_{\text{RMS}} = \frac{1}{N}\sqrt{\sum^{N}(x_i - z_i)^2} \tag{5.1}$$

$$\text{Mean Absolute Error} \qquad e_{\text{MAE}} = \frac{1}{N}\sum^{N}|x_i - z_i|\,. \tag{5.2}$$

Here, x_i represents the calculated quantity over any range of locations or times, and z_i represents the measured quantity (or known quantity, if available) at the same location and time. The domain of interest is designated by N. The difference between the two is that the root mean squared error formulation will become larger if there are outliers, whereas the mean absolute error formulation will not be a influenced by rogue values. Realize that the quantity e can represent either an error (difference between calculated and known heat rates) or a residual (difference between measured and calculated temperatures).

This quantification contains information as to the level of error in the estimate. There are primarily two types of error found here: variance error and bias (deterministic) error. The difference between the two types can best be explained graphically. Fig. 5.1 shows how a biased estimator differs from an estimator which amplifies the variance error. Unfortunately, the root mean squared error and mean absolute error can not resolve this difference except in extreme cases. Therefore, we must examine the estimates and their residuals to learn which type of estimator our methods are. Note that most estimators contain a combination of both types of errors.

5.2 Test Cases

Aside from the use of experimental data for validation, the methods will be benchmarked against a predetermined heat flux. As described in Fig. 1.4, simulated data will be obtained from the transient heat rates of the test cases illustrated in Fig. 5.2. The finite element code EAL was used to generate the simulated measurements from standard test cases. An attempt was made to insure that the conduction solution method was not equivalent to the method used in the estimation routines. Even though the model was the same, the number of nodes was increased, the resolution of the property data was increased and the integration

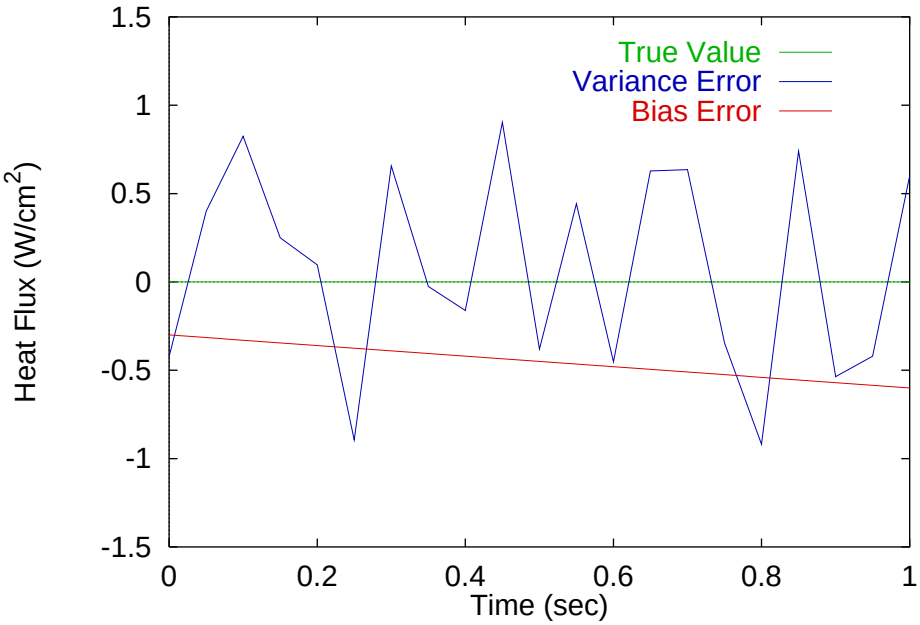

Fig. 5.1 **The two types of errors shown depict how biased estimators might "miss" the true value by some consistent offset, where as another estimator might simply amplify noise and increase the variance error.**

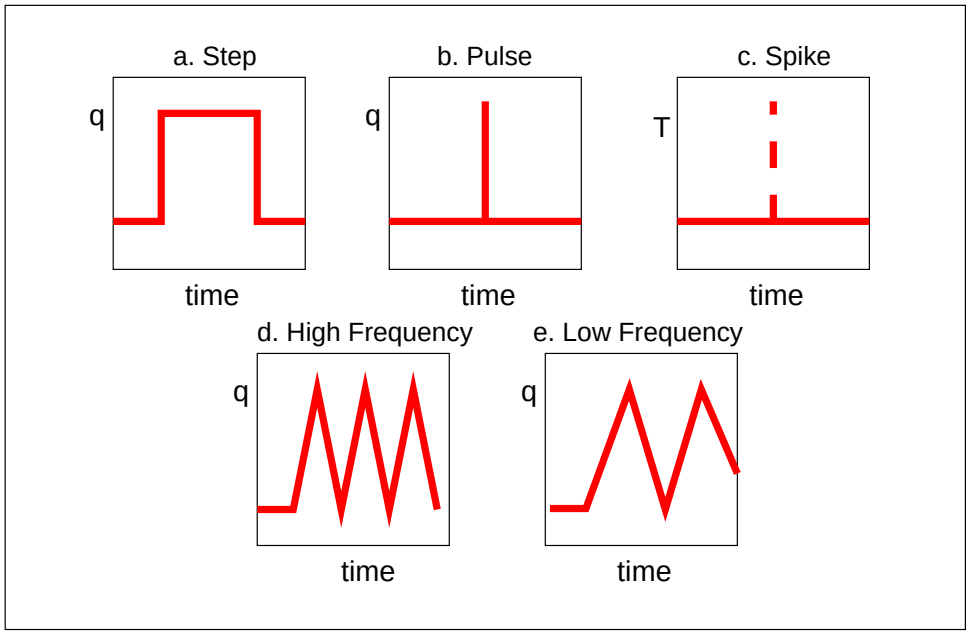

Fig. 5.2 **Different test cases used for evaluation of one-dimensional methods.**

time step was increased. For a single case, the finite element forward solution was verified using a finite difference code. (See Appendix C for a description of the conduction solution methods and their validation.)

It has become standard practice within the inverse community, as outlined by Beck (1979), to use test cases in which a heat flux function is defined and then estimated using simulated data to evaluate the behavior of a method. Murio (1985) also suggests that one of the advantages of evaluating methods in this way is that it allows for appropriate evaluation of parameters involved in the estimation such as the level and type of regularization.

Note that the parameters (magnitudes, sample rate, etc.) dealing with the predetermined data were selected by attempting to mimic the conditions of an actual experiment to be discussed in Chapter 6. The conditions from the experiment that were of concern are given in Table 5.1. The five specific test cases, to be discussed in the following sections and pictorally described in Fig. 5.2, were generated based on these parameters. Furthermore, noise was added to each case as an enhancement to the simulation.

5.2.1 Simulated Noise

When temperature measurements are recorded, there is usually some amount of random noise in the signal that does not represent the actual temperature signature. Although small, this noise is amplified by estimation techniques and becomes significant. Therefore, the temperature data that are created using the predetermined fluxes have normally distributed random errors with a variance of 0.5 K added. This noise addition process is performed after the exact temperature signature is found thus simulating additive errors. This procedure

Table 5.1 Experimental parameters from Run 43 and Run 85 used for generation of data for test cases.

	Run 43	Run 85
Start Time	0.00005 sec	0 sec
End Time	0.011 sec	0.01995 sec
Time Step	0.00005 sec	0.00005 sec
Number of Time Steps	211	400
Sample Frequency	20 kHz	20 kHz
Material	Pyrex	Pyrex
Initial Temperature	528°F	542°F
Maximum Temperature	1201°F	979°F

was applied to all test cases.

5.2.2 Step Flux

In a discrete sense, the step change in flux represents a large, sudden change in the heat rate as shown in Fig. 5.2a. This type of flux can be found immediately following a 3 msec "start up" period when the flow first impinges the model. Since this type of heat flux is usually found at the beginning of experiments, it is important to determine what effect this characteristic might have on downstream estimates as well as determine the behavior of the estimates during a constant flux after the step. The hypothetical flux is zero until 3 msec where it jumps to 1135 W/cm^2 (1000 $Btu/ft^2 sec$). The flux is then set back to zero at 7 msec. The features highlighted by these data include large temperature changes and constant flux conditions.

5.2.3 Pulse Flux

This type of test case, depicted in Fig. 5.2b, does not model any specific physical phenomenon, but should provide insight into the behavior of the estimation routines at large singular fluctuations and the level of bias added to the solution. The data consist of a single heat flux data point of 1135 W/cm^2 (1000 $Btu/ft^2 sec$) midway through the data region and a zero flux otherwise.

5.2.4 Temperature Spike

We can simulate experimental errors such as electrical pulses recorded by acquisition facilities that fall outside the normal range of experimental noise. An example of this behavior was recorded at GASL Trucco and Tamango (1990), when Schlieren photographs caused an errant spike with an order of magnitude greater than the nominal experimental noise. For this test case (Fig. 5.2c), the temperature produced by the pulse is treated as an error in temperature measurement. In other words, the flux is zero but we retain the temperature signature at the point of the spike as if the flux did exist. With this test case, the effects of anomalistic measurements on the data reduction can be studied as well as the effects of measurement noise.

5.2.5 High Frequency Oscillating Flux

Because of the unsteady nature of the impinging jet, the heat rate can fluctuate rapidly with a large amplitude. Therefore a hypothetical flux with a sinusoidal distribution will be examined. The Nyquist assumption limits the highest frequency we can recover to be half the sampling rate or $f_n = 10\,\text{kHz}$ in this case. As illustrated by Fig. 5.2d and e, two distributions are examined, $f_n/2 = 5000\,\text{Hz}$ and $f_n/8 = 1250\,\text{Hz}$, which represent one half and one eighth the Nyquist frequency respectively.

Chapter 6

Unsteady Analysis

The evaluation of the methods and of the proposed technology was divided into two parts. The first part involves examining the treatment of unsteady data; the second part involves lateral conduction effects, or the nonuniformity of imposed fluxes. For isolation of the unsteady heating rates one-dimensional conduction analysis is assumed. This approach will also validate the proposed technology by verifying its ability to estimate unsteady heating rates as accurately (and in most cases better) than the established methods. The nonuniform side of the problem will be addressed in Chapter 7.

The eleven different one-dimensional estimation methods were tested against simulated measurement data created from predetermined heat fluxes. The test cases were generated as described in Section 5.2 and mimic actual measurements obtained from NASA LaRC. The test data were modeled after two particular experiments because the heating rates were presumed to be highly dynamic (unsteady) and relatively large (see Fig. 6.1 for the measured temperature response of the two experiments). These data provide a baseline from which the test data were derived. Once the test cases were analyzed, the actual experimental fluxes were estimated and conclusions drawn based on the performance as exhibited on the test cases.

6.1 Analysis of Test Cases

The results of estimates of the test cases have been arranged first to provide an average quantification of the overall performance of the each method, and then to accentuate common features found in most estimation procedures. These features are characteristics which

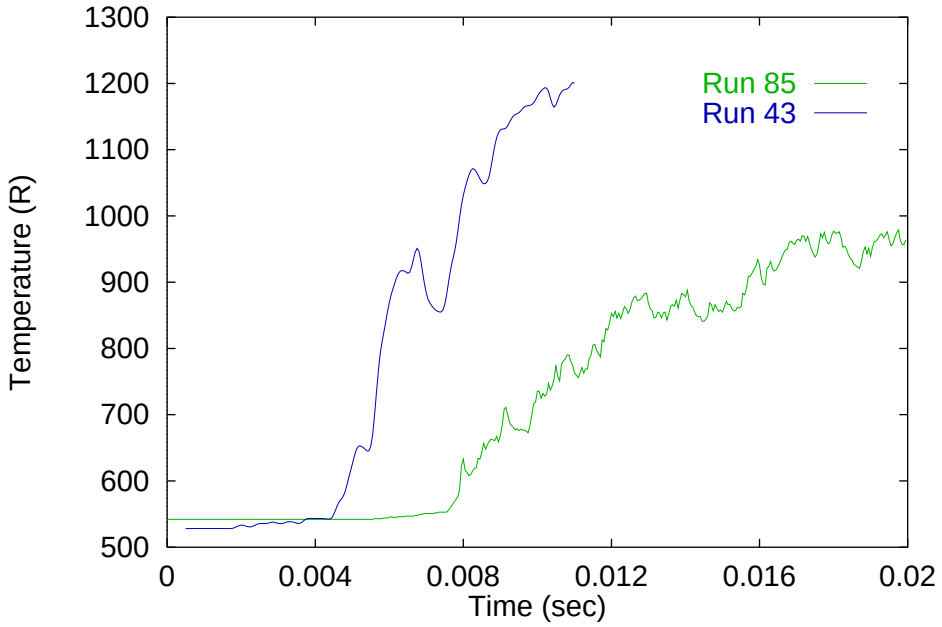

Fig. 6.1 Unsteady nature of the measured temperatures of Run 85 and Run 43 (Calspan).

contribute to the failure of a particular method to describe adequately the heating rate. In each case, the appropriate inverse method is compared with the offending methods to demonstrate the robustness of the preferred methodology.

6.1.1 Global Behavior

A qualitative comparison of the performance of the estimates is obtained by calculating the statistical errors of the estimates and the residuals as described in Section 5.1. These values were calculated from the results of each test case using eleven different methods studied. Table 6.1 and 6.2 show the RMS errors of estimated heat flux and residuals respectively; the corresponding mean absolute values are shown in Table 6.3 and 6.4.

The first important trend to notice is the relationship between the errors and the residuals. It is encouraging to realize that there is a strong indication as to the linearity of the relationship for all types of data. In other words, the error is related to the residuals by some constant factor. This suggests that when real data are examined, the residual can lead us to a good understanding of the error in the estimate. This is important since, for real data, we do not know the actual heat flux and therefore, do not know the error of our

Table 6.1 Root Mean Squared Error of the Estimated Heat Flux

Heat Flux (W/cm^2)							
Method			Test Cases				
0 - without noise						Oscillations	
n - with noise			Step	Pulse	Spike	$f_N/2$	$f_N/8$
Class 1	*Cook-Felderman*	0	7.98	6.72	2.90	30.27	9.60
		n	8.04	7.00	3.14	30.38	9.70
	Kendall-Dixon	0	9.17	5.17	2.80	21.90	18.84
		n	9.17	5.17	2.80	21.90	18.84
	Diller	0	4.79	3.16	3.05	12.67	3.82
		n	4.84	3.13	3.04	12.64	3.89
Class 2	*Finite-Difference*	0	2.20	1.30	2.95	5.80	2.39
		n	2.64	1.53	3.19	5.98	2.75
	Simple-Implicit	0	2.26	2.39	1.45	8.94	2.73
		n	2.61	2.42	1.62	9.04	2.95
	Rae-Taubee	0	4.73	1.65	2.54	7.26	3.13
		n	4.91	1.81	2.76	7.39	3.37
Class 3	*Constant-Specification*	0	4.44	4.25	2.81	21.28	5.01
		n	4.46	4.24	2.80	21.28	5.03
	Linear-Specification	0	4.11	4.19	2.60	20.72	2.89
		n	4.12	4.18	2.59	20.71	2.92
	Exact-Matching	0	0.34	0.43	5.91	1.15	0.26
		n	2.34	1.65	6.28	2.35	2.09
	Regularization	0	1.03	0.65	0.52	3.59	0.46
		n	1.90	1.33	1.38	3.82	1.58
	Explicit-Regularization	0	0.48	0.17	0.93	1.13	0.17
		n	2.10	1.48	5.75	2.22	1.86

Table 6.2 Root Mean Squared Residual of the Calculated Temperatures

Temperature (K)							
Method			Test Cases				
0 - without noise						Oscillations	
n - with noise			Step	Pulse	Spike	$f_N/2$	$f_N/8$
Class 1	*Cook-Felderman*	0	1.08	0.10	2.89	0.48	0.51
		n	1.08	0.10	2.90	0.48	0.51
	Kendall-Dixon	0	1.19	0.38	2.88	0.97	1.80
		n	1.20	0.40	2.88	0.98	1.81
	Diller	0	1.03	0.14	2.80	0.55	0.48
		n	1.03	0.16	2.81	0.55	0.48
Class 2	*Finite-Difference*	0	0.48	0.09	2.87	0.32	0.27
		n	0.48	0.10	2.87	0.32	0.27
	Simple-Implicit	0	0.25	0.15	2.81	0.44	0.33
		n	0.25	0.16	2.82	0.44	0.33
	Rae-Taubee	0	1.07	0.11	2.86	0.64	0.63
		n	1.07	0.11	2.86	0.64	0.63
Class 3	*Constant-Specification*	0	0.29	0.23	2.73	0.84	0.45
		n	0.29	0.24	2.74	0.85	0.46
	Linear-Specification	0	0.22	0.22	2.74	0.82	0.28
		n	0.23	0.23	2.74	0.83	0.28
	Exact-Matching	0	0.02	0.02	2.93	0.04	0.02
		n	0.02	0.03	2.94	0.05	0.02
	Regularization	0	0.03	0.04	0.31	0.11	0.02
		n	0.04	0.04	0.32	0.12	0.03
	Explicit-Regularization	0	0.03	0.03	0.06	0.11	0.04
		n	0.03	0.04	0.17	0.11	0.04

Table 6.3 Mean Absolute Error of the Estimated Heat Flux

Heat Flux (W/cm^2)							
Method			Test Cases				
0 - without noise						Oscillations	
n - with noise			Step	Pulse	Spike	$f_N/2$	$f_N/8$
Class 1	*Cook-Felderman*	0	45.55	10.85	5.44	350.38	100.51
		n	51.87	21.48	15.44	354.59	104.61
	Kendall-Dixon	0	72.56	10.08	4.53	185.35	195.22
		n	72.94	10.65	5.04	185.51	195.43
	Diller	0	39.47	5.17	6.35	142.55	40.10
		n	42.02	10.33	11.16	143.65	42.14
Class 2	*Finite-Difference*	0	12.39	2.37	5.21	68.11	25.15
		n	23.17	12.32	15.12	71.78	30.23
	Simple-Implicit	0	4.22	3.70	3.44	105.29	29.03
		n	16.76	10.90	10.19	108.13	32.39
	Rae-Taubee	0	48.95	2.48	4.96	83.14	29.63
		n	52.99	12.08	14.13	86.74	34.91
Class 3	*Constant-Specification*	0	16.13	7.57	5.61	208.03	53.54
		n	18.49	9.99	7.77	208.89	54.32
	Linear-Specification	0	12.32	7.66	5.20	193.73	30.70
		n	15.19	9.65	6.88	194.39	31.78
	Exact-Matching	0	0.85	0.75	11.18	12.60	2.71
		n	25.32	17.56	27.24	26.10	22.92
	Regularization	0	2.53	1.38	1.06	41.48	4.79
		n	19.12	14.47	14.56	45.19	17.63
	Explicit-Regularization	0	1.13	0.31	0.81	10.82	1.73
		n	22.55	16.17	13.83	24.57	20.46

Table 6.4 Mean Absolute Residual of the Calculated Temperatures

Temperature (K)							
Method			Test Cases				
0 - without noise						Oscillations	
n - with noise			Step	Pulse	Spike	$f_N/2$	$f_N/8$
Class 1	*Cook-Felderman*	0	9.42	0.59	3.29	4.69	5.22
		n	9.50	0.77	3.47	4.81	5.31
	Kendall-Dixon	0	12.04	1.22	3.21	10.68	18.74
		n	12.23	1.80	3.87	10.91	18.94
	Diller	0	9.10	0.43	3.02	5.43	4.59
		n	9.26	0.77	3.48	5.63	4.77
Class 2	*Finite-Difference*	0	4.50	0.25	3.24	2.94	2.74
		n	4.60	0.49	3.45	3.06	2.84
	Simple-Implicit	0	1.50	0.33	3.05	3.78	3.24
		n	1.64	0.69	3.43	3.97	3.39
	Rae-Taubee	0	9.58	0.28	3.13	6.37	6.10
		n	9.68	0.56	3.39	6.50	6.23
Class 3	*Constant-Specification*	0	1.56	0.48	2.99	9.72	4.77
		n	1.81	1.09	3.59	9.95	4.99
	Linear-Specification	0	0.79	0.48	3.00	9.56	2.92
		n	1.25	1.08	3.61	9.77	3.12
	Exact-Matching	0	0.11	0.06	3.03	0.42	0.20
		n	0.14	0.14	3.11	0.46	0.24
	Regularization	0	0.16	0.09	0.32	1.24	0.21
		n	0.29	0.23	0.43	1.30	0.30
	Explicit-Regularization	0	0.14	0.07	0.05	1.26	0.38
		n	0.22	0.19	0.13	1.31	0.44

estimate.

Secondly, we also notice that Class 1 methods are generally not preferable because the error as a whole is larger than the other two classes. Furthermore, we notice that the errors associated with the Class 2 techniques, are larger still than the errors of the Class 3 methods. Never the less, there are isolated incidents where a particular method performs particularly well given a certain type of data.

One obvious example of a method that is particularly suited for certain types of data is the *Kendall-Dixon* method. When the data are known to be fairly constant or when large errors are to be damped, the wide differencing scheme results in very smooth estimates. This feature is apparent in the temperature spike test case. The *Diller* method also provides generous smoothing though not to the extent of the *Kendall-Dixon* method. Note however, the *Diller* method does not suffer from the overwhelming additional bias that the *Kendall-Dixon* method exhibits on the other test cases. Specific characteristics of the methods' performance are illustrated below.

6.1.2 Noise Amplification

The first feature of estimation routines is the amplification of noisy data. It is well understood that most Class 1 techniques exhibit this behavior (Ehrich, 1954Diller and Kidd, 1997). For the test cases examined, the added noise was on the order of 0.5°C which can be considered typical for thin film temperature sensors. Fig. 6.2 shows the simulated temperature measurements for the step flux case with the added noise. Even though the noise is barely evident through visual inspection, the estimate amplifies this virtually imperceptible fluctuation into an obvious error. The estimates to the noisy step case are shown for the *Cook-Felderman* (Fig. 6.3) and the *Finite-Difference* (Fig. 6.4) respectively. The amplification of noise tends to occur in all Class 1 and Class 2 techniques that do not introduce some level of bias. The *Kendall-Dixon* and all inverse techniques use added bias to quiet the amplification of noise. To see how the treatment of noisy data differs, we turn to the spike test case and examine the time where nothing is happening (prior to the temperature spike). We expect a zero heat flux estimate except that there is added noise which will become amplified. Fig. 6.5 shows the results of *Simple-Implicit* method prior to the spike, which does not contain bias, along with the *Kendall-Dixon* and the *Regularization* which smooth the estimate.

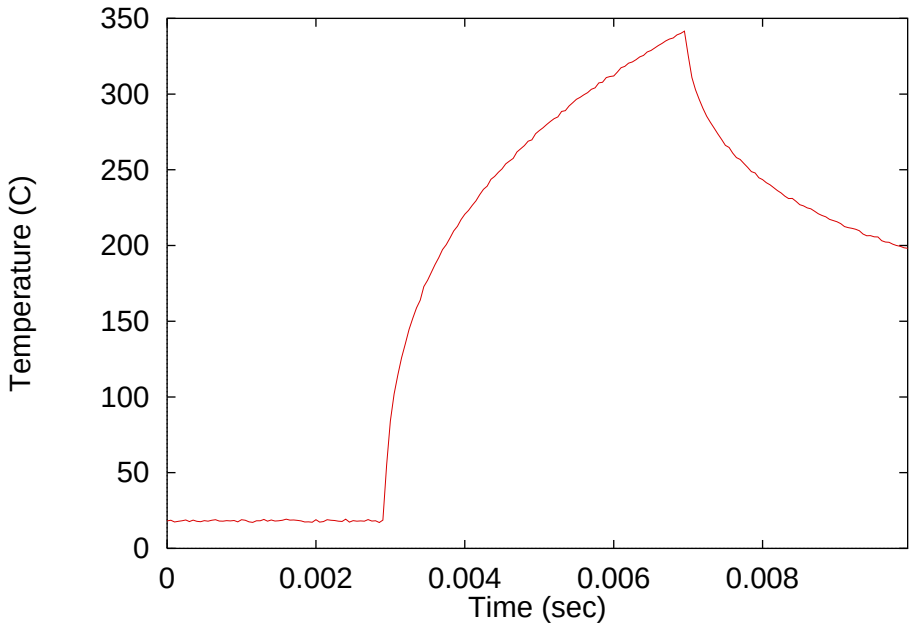

Fig. 6.2 **Simulated data for the Step Flux Test Case with added noise.**

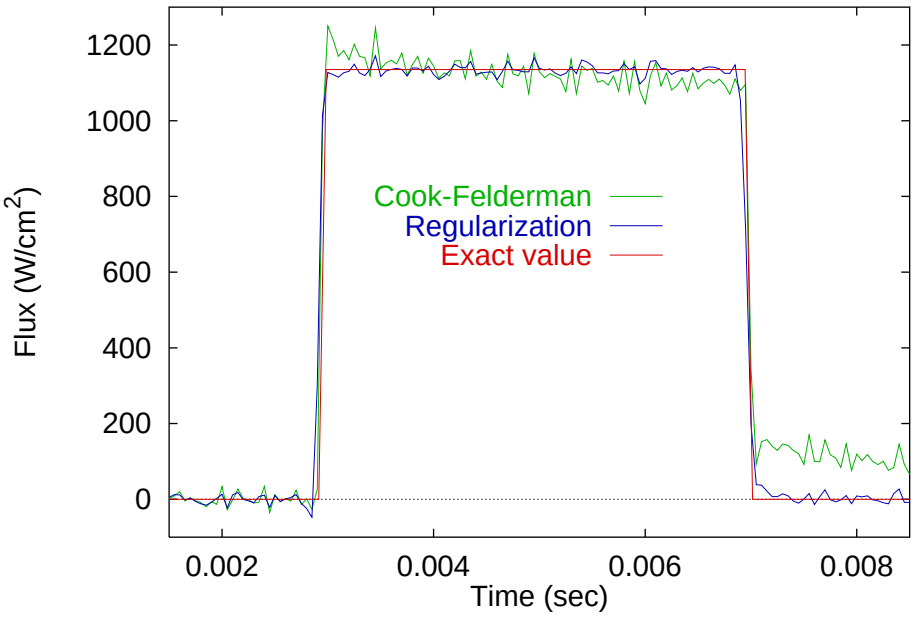

Fig. 6.3 ***Cook-Felderman*** **and** ***Regularization*** **flux estimates for the Step Flux Case with added noise.**

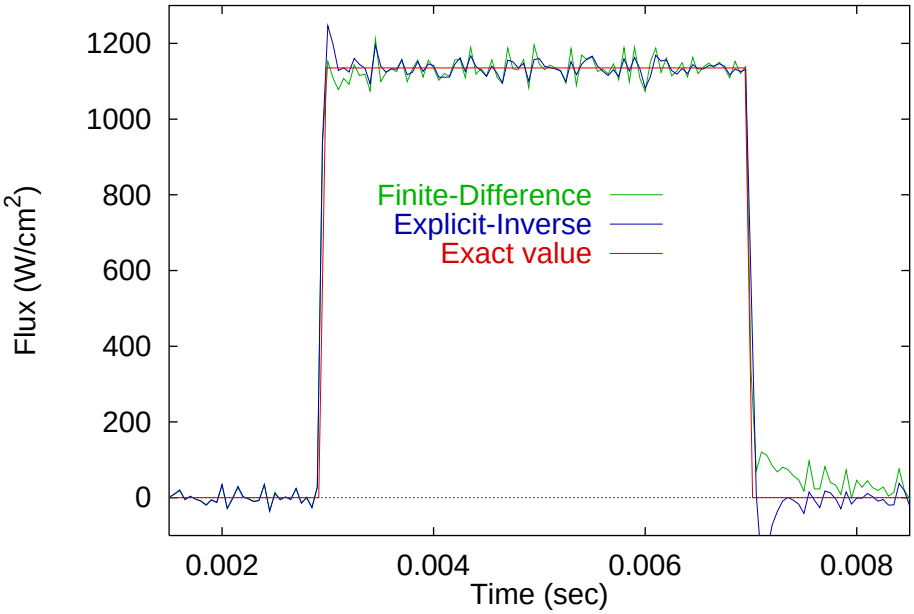

Fig. 6.4 ***Finite-Difference*** **and** ***Explicit-Regularization*** **flux estimates for the Step Flux Case with added noise.**

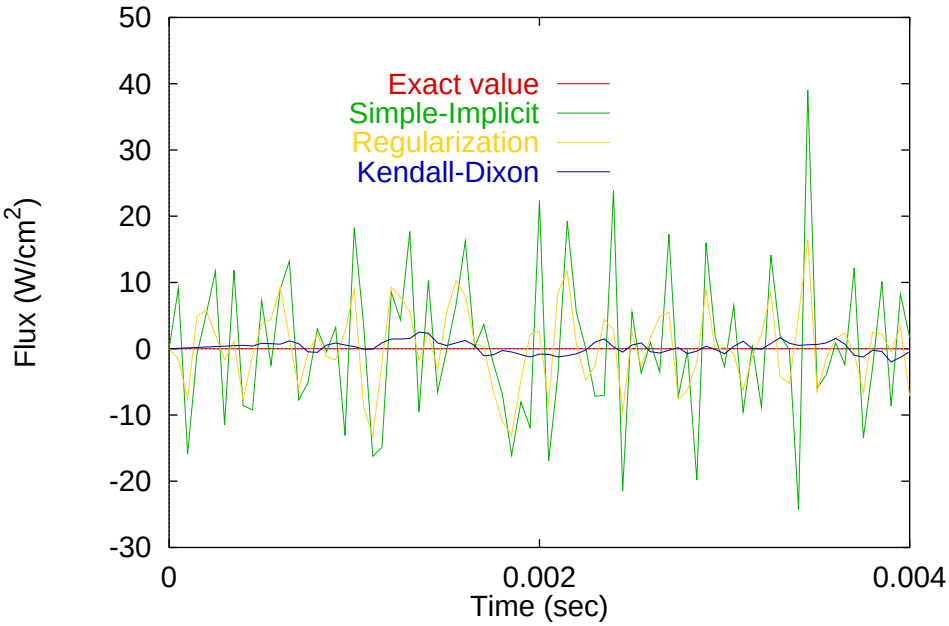

Fig. 6.5 ***Kendall-Dixon***, ***Simple-Implicit*** **and** ***Regularization*** **flux estimates from noisy data with zero mean (Temperature Spike Test Case before the actual Spike).**

6.1.3 Solution Damping and Lagging/Leading

Generally, estimation procedures do not dampen the solution. As previously mentioned, the solution is relatively sensitive to measurement errors and uncertainties, which causes the estimates to become exaggerated. This instability, however, has lead to some techniques introducing artificial biasing which is designed to control the amplification of uncertainties. In some cases, the bias results in a damped solution. The canonical example of over damping the solution can be seen with the *Kendall-Dixon* technique. Because of its wide differencing scheme, the estimate over several time steps is effectively average. While this could be appropriate for relatively steady situations, high frequency oscillations can be lost as demonstrated by the estimates in Fig. 6.6. Initially notice how the estimates of the high frequency and the low frequency test cases are inappropriately damped. Now realize that the lower frequency case begins to realize some of the features of the test flux. Since the differencing scheme spans eight time steps, the method will respond favorably (although the damping will still predominate) once the frequency drops below one eighth of the Nyquist frequency.

Because the inverse techniques also employ added bias to stabilize the solution and to reduce the amplification of uncertainties, they are susceptible to damped solutions as well. However, the level of bias is carefully controlled so that the deterministic error is a minimum and very little damping occurs as seen in Fig. 6.7.

A solution that lags or leads the true flux is often related to a solution that is damped. It was discussed in Section 6.1.2 how the typical treatment for noise was to introduce bias. Unfortunately, this has a side effect of not only damping the estimate but also causing the estimate to lead the actual solution. Typically, the addition of bias involves including information about future measurements and in some cases previous measurements. This type of information may influence the current estimate to behave like estimates at either future or previous times, thereby causing a leading or a lagging effect. This feature can be seen predominately in the *Kendall-Dixon* method which employs excessive smoothing through a forward looking and backward looking biasing scheme of eight time steps. The pulse test case demonstrates this feature in Fig. 6.8.

The pulse case was designed to test for biasing errors in the solution. Therefore, if we examine Fig. 6.9 we can see that the *Linear-Specification* method is fairly biased compared to

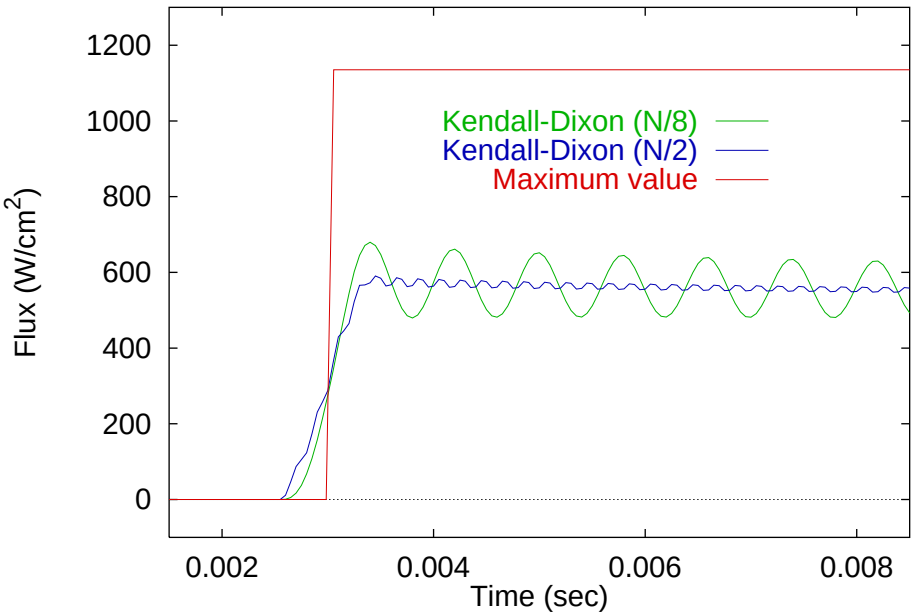

Fig. 6.6 **The *Kendall-Dixon* flux estimates for the Oscillating Flux Test Cases.**

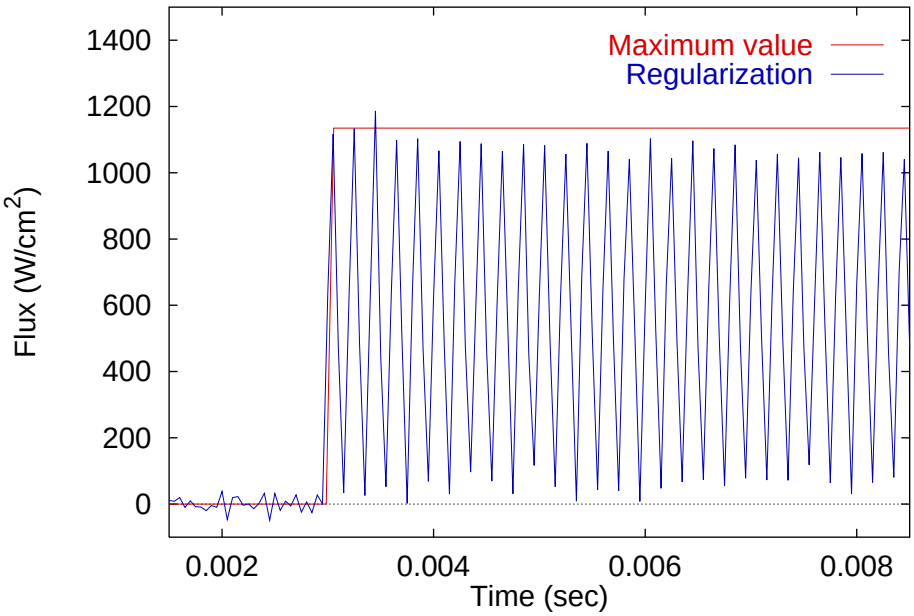

Fig. 6.7 **The *Regularization* flux estimates for the High Frequency Oscillating Flux Test Case.**

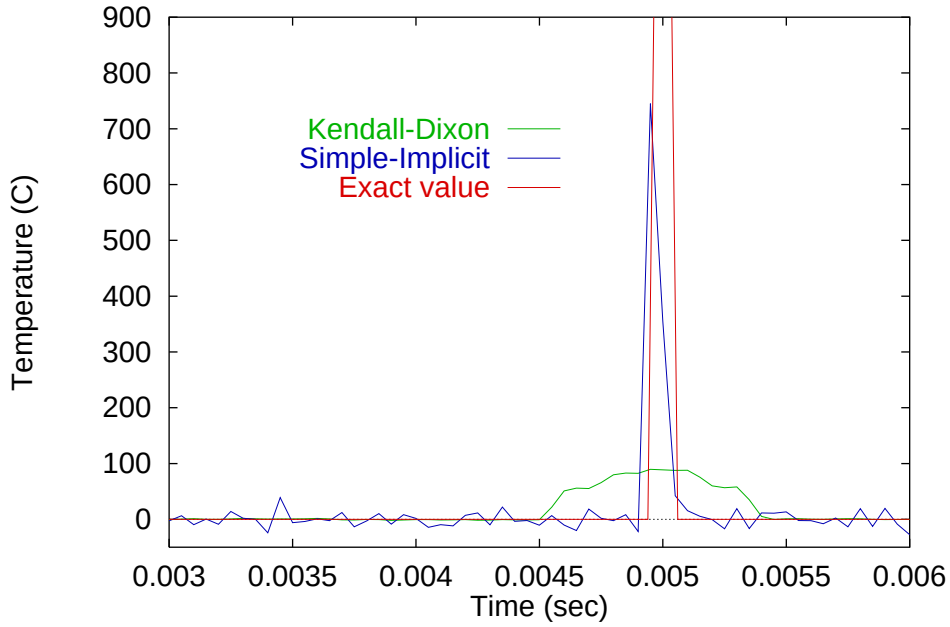

Fig. 6.8 The Leading and Lagging effects of the *Kendall-Dixon* and the *Simple-Implicit* methods for the Pulse Flux Test Case with added noise.

the *Diller* case. Likewise, all Class 2 cases will appear biased as a result of their gridding issues discussed in the following section (Section 6.1.4). On the other hand, Class 1 techniques come with varying degrees of imposed bias. For example, the *Cook-Felderman* method is designed without additional bias, whereas *Kendall-Dixon* method typically contains too much imposed bias. Finally, Class 3 methods typically demonstrate bias except where the amount can be controlled such as in the *Regularization* and the *Explicit-Regularization* methods. Note, however, that the *Exact-Matching* method has no inherent bias added.

Another device used to determine the type and amount of bias is to examine the residuals. The residuals are calculated by finding the temperature response to the estimated flux. These temperatures are then compared to the original temperature measurements. Notice how Fig. 6.10 demonstrates the inherent lead/lag of the step flux in the solution for the *Kendall-Dixon* method and the bias due to the lack of temperature dependent properties in the *Diller* method for the step flux. The bias associated with the *Rae-Taubee* residual is due to the gridding issues (that will be discussed in Section 6.1.4). A close look at the residuals for the *Explicit-Regularization* and the *Simple-Implicit* case show that both are relatively close to zero. The inverse approach, however, does a better job of reducing

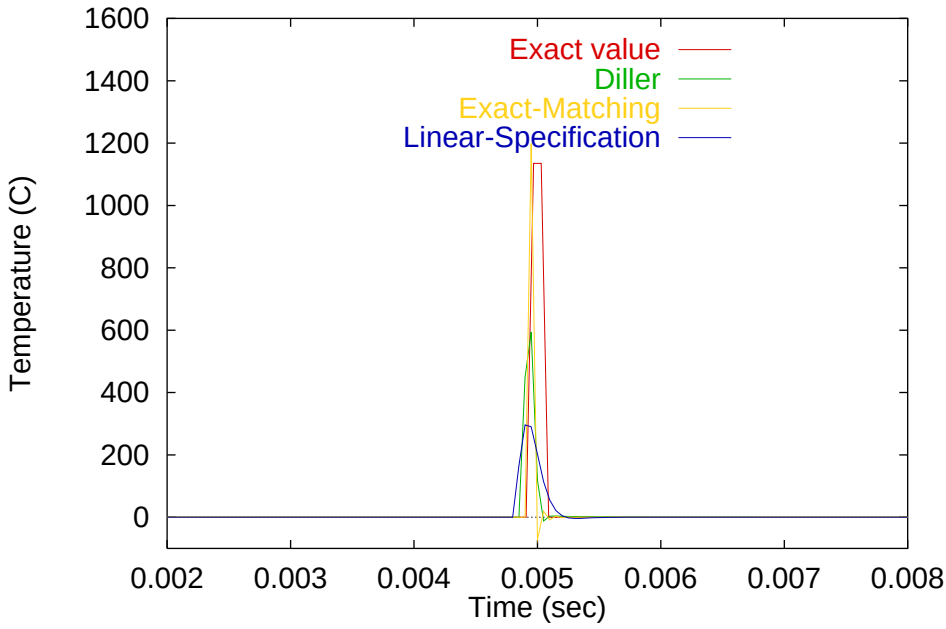

Fig. 6.9 The *Diller*, *Linear-Specification* and *Exact-Matching* flux estimates for the Pulse Flux Test Case.

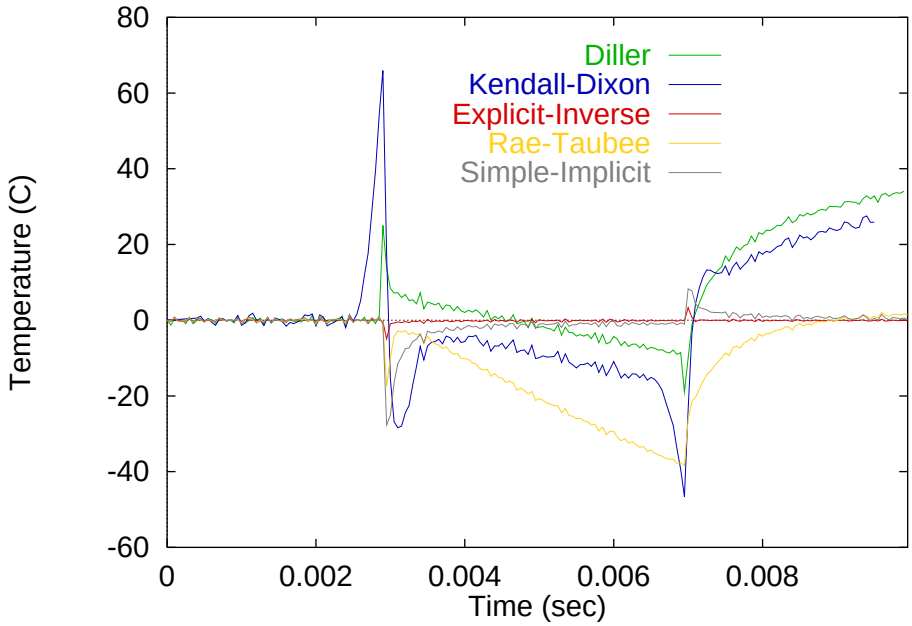

Fig. 6.10 Residuals for the *Kendall-Dixon*, *Diller*, *Explicit-Regularization*, *Rae-Taubee* and *Simple-Implicit* methods for the Step Flux Test Case.

the residual at the singularity and does not tend to introduce deterministic error as the *Simple-Implicit* does.

Additional information can be gleaned from examination of the other data sets. Most methods exhibit different behavior when the flux oscillates dynamically. The residuals given for the Low Frequency Test Case in Fig. 6.11 show that the *Diller* method, while it handles noisy data well, introduces a bias by lagging the actual flux. (Residuals for the high frequency case show the same information except the residuals are larger; the graph is more difficult to read.)

6.1.4 Gridding Issues

Naturally, only Class 2 techniques will suffer from inappropriate gridding. Despite the fact that inverse approaches can use a numerical conduction solution as part of their overall solution, it will be shown that the concerns do not affect the performance of inverse methods.

At this point, let us assume that grid independence has been achieved for *all* numerically based estimation routines in reference to the temperature solution. In other words, the temperature solution can be obtained accurately. Unfortunately, this is not the only issue faced by numerical approaches.

Surface Derivative Resolution

Recall that Class 2 techniques require a calculation of the temperature derivative at the surface. Also recall that this gradient is extremely steep because the heat fluxes of interest are large. As a result, the finite difference calculation becomes extremely sensitive to errors in the temperature calculation. Even when the temperature is said to be known accurately, small fluctuations in the temperature measurements can result in large deviations in the gradient. Furthermore, the gradient can not be accurately modeled with a finite difference approximation. Fig. 6.12 shows the *Finite-Difference* method's estimation of the step flux. The initial jump in the estimate at the inauguration of the step is a result of the sensitivity of the system. After the initial rise, the estimate settles to a value *less* than the expected value. At early times, the gridding offered to resolve the flux is coarse compared to the penetration depth of the flux which is given as a function of time

$$\eta_p = \sqrt{\alpha_o \Delta t}\,. \tag{6.1}$$

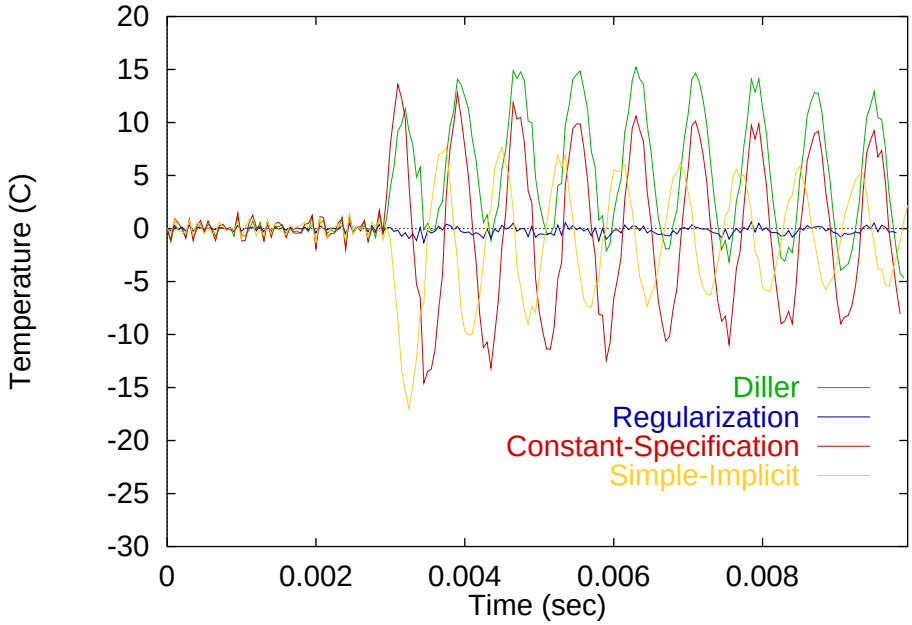

Fig. 6.11 **Residuals for the *Diller*, *Regularization*, *Constant-Specification* and *Simple-Implicit* methods for the Low Frequency Oscillating Flux Test Case.**

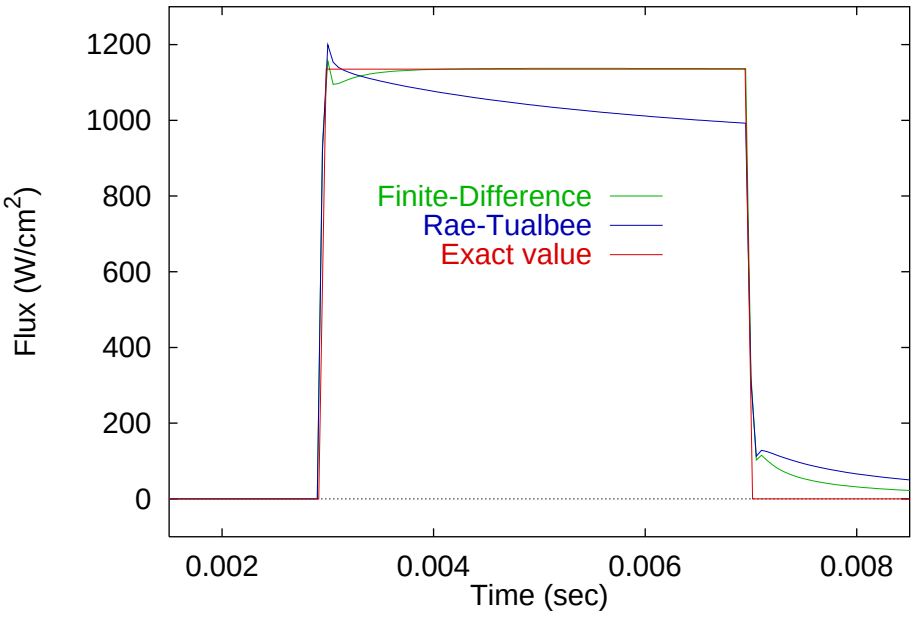

Fig. 6.12 **The *Finite-Difference* method's inadequate gridding effects at early times, and the *Rae-Taubee* method's lack of grid resolution at later times for the Square Flux Test Case.**

In other words, when the grid spacing is fixed as in the *Finite-Difference* and *Simple-Implicit* methods, the penetration of the flux for a single time step may be less than the actual node spacing. (Remember that the temperature solution is assumed to be converged.) Using this argument, how could the finite difference formulation between adjacent nodes possibly represent the derivative at a node?

For the conditions given by the step flux, a nodal count of 50 resulted in a converged temperature solution. If the penetration depth of the experiment is given as $\eta_f = 2\sqrt{\alpha_o 200\Delta t} \approx 0.06$cm, the ratio of the penetration depth of a single time step with respect to the nodal spacing then becomes

$$\frac{\eta_p}{\Delta x} = \frac{50\sqrt{\alpha_o \Delta t}}{0.06\text{cm}} \approx 1.7 \tag{6.2}$$

This value represents the distance to which the current heat flux is felt within the surface control volume. Additionally realize that at 1.7 control volumes there is a minscule response. It becomes immediately clear that the response of the first inbound node does not reflect the actual input on the surface since the penetration depth is so small compared to the node spacing.

It is suggested here, that the number of nodes required to accurately characterize the flux can be determined from the number of nodes needed to resolve the temperature distribution (N). The rule of thumb requires N nodes to be placed within the penetration depth of a single time step. Thus, for a fixed grid with $N = 50$, the number of computational nodes to resolve the flux becomes the number of nodes per single time step penetration depth times the experimental penetration depth

$$\#\text{ of nodes} = \frac{N}{\eta_p}\eta_f = 1,500\,. \tag{6.3}$$

Now realize that the integration time steps required to resolve this tightly packed nodal array is inversely proportional to the square of the node spacing. Given these parameters, the one-dimensional conduction problem becomes unmanageable. Due to time constraints and lack of computer resources, a rigorous demonstration of this concept was not performed. However, a limited number of trial runs indicated preliminary verification.

For the forward conduction solutions reported here, the number of nodes was quadrupled to insure temperature convergence and to attempt to model the surface gradients. As we will find, this inordinately miniscule grid spacing will be inadequate to fully capture the surface gradient as suggested.

Bias Due to Variable Grid

To compensate for the extreme demand for small nodal spacing, the *Rae-Taubee* method was designed to follow the penetration of the heat flux. Although this concept greatly enhances the accuracy of the estimates at early times, the grid spacing becomes stretched at later times. In other words, the same number of nodes are continually shifted to cover a more area. At the end of the estimation, the spacing is identical to that of other Class 2 methods that use a fixed grid. From Fig. 6.12, we can see how the *Rae-Taubee* even tends to introduce a bias at later times. This effect was also noted by Holden et al. (1991).

6.1.5 Temperature Dependent Properties

It is assumed that the numerical techniques examined here are capable of solving the non-linear conduction problem. Therefore, Class 2 techniques are not affected by large temperature variations that may cause property data to change within the analysis. Class 1 techniques, however, by nature can not account for temperature dependent properties directly. Through a correction factor, the estimates can be improved, though.

The correction factor is found by calculating a temperature response to a known heating rate, assuming constant thermal properties. The solution which accounts for the variation of the properties is also found. The correction is then found by comparing the two temperature responses for the particular case. The problem with this method is that the correction is then valid only for the particular case for which it was calculated. To suggest that any given correction is valid for a wide range of problems is questionable. A general discussion of the behavior of the correction is provided in Appendix D. The correction used for this study is considered to be established and widely applicable (Hollis, 1995). Never the less, Fig. 6.13 suggests that the correction is not robust for any of the Class 1 methods. As the temperature rises, the correction looses its effectiveness.

6.1.6 Flux Modeling Assumption

Some estimation methods assume that the flux is piecewise constant over time. The finite element code used to generate the temperature data, however, more accurately assumes a piecewise linear distribution between time steps (see Fig. 6.14). Even though the integral of both histories may be the same, the resulting temperature distributions will not be

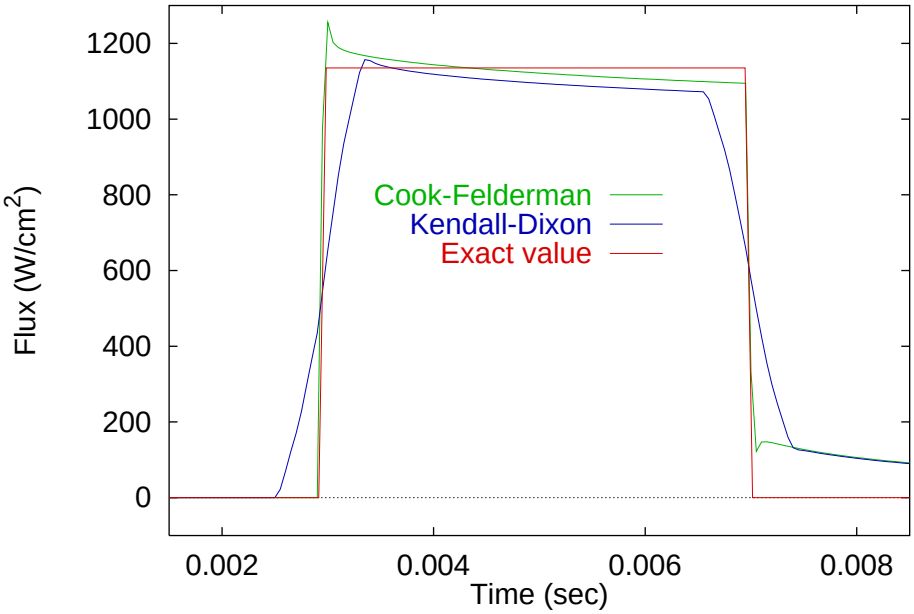

Fig. 6.13 **The effects of the constant properties with a correction term on the heat flux estimates using the *Cook-Felderman* and *Kendall-Dixon* methods and the Step Flux Test Case.**

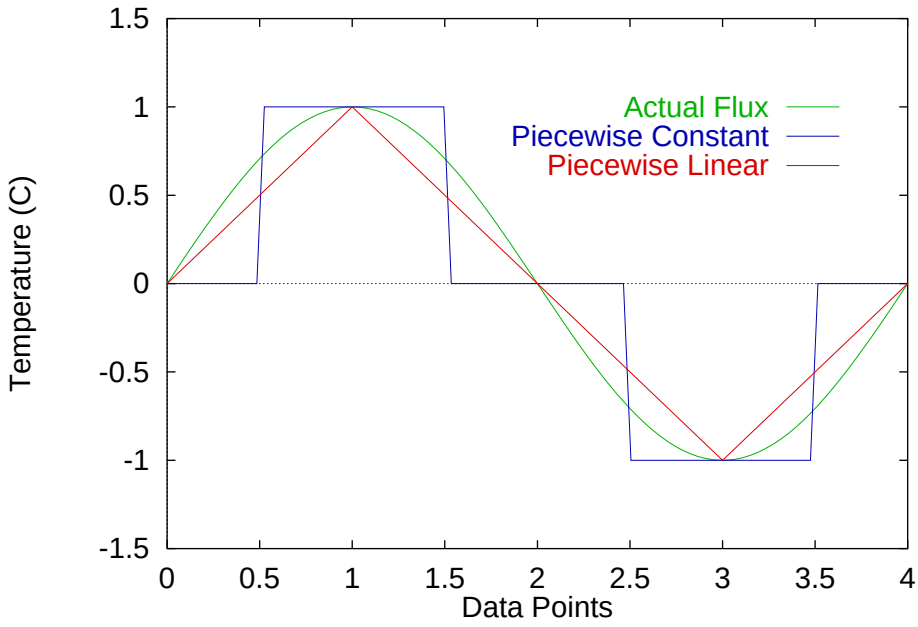

Fig. 6.14 **Piecewise constant assumption compared to the piecewise linear assumption that result in different temperature responses.**

similar especially at the surface where the measurements occur. This influence, however, is less apparent at lower frequencies. Also note that, generally, the effects of different functional forms of the flux can not be separated from the bias introduced by gridding issues discussed in Section 6.1.4. Furthermore, it would be equally difficult to calculate the relative influence if each effect unless one effect could be eliminated. This analysis was not performed, therefore, evidence of flux modeling issues on the estimate is inconclusive.

6.2 Class 3 Features

For the most part, the inverse solutions typically do not suffer from the ailments of the Class 1 and Class 2 techniques. However, these methods generally have the tendency to amplify noisy data. This feature, though, is mitigated by the addition of biasing. It was shown that methods which add bias, have the tendency to lag and/or lead. Ideally the amount of bias should be controllable to obtain accurate solutions while assuaging noise effects.

The *Function-Specification* methods (*Exact-Matching*, *Constant-Specification* and *Linear-Specification*) do not allow infinitely variable (adjustable) amount of added bias. We are limited by the order and the complexity of the function that is to be applied. Note that *Exact-Matching* uses no bias because the "function" that it employs is simply the current estimate. For the other function specification methods to work we must look into the future a certain number of time steps. Therefore, the minimum number of future times was used to achieve the minimum level of bias (one future time for *Constant-Specification* and 2 future times for *Linear-Specification*). Fig. 6.15 shows how the amount of bias affects the estimate of the step test case. Notice how the *Exact-Matching* case over estimates the jumps due to the lack of biasing. Of the biased methods, note that the *Linear-Specification* case provides a better estimate (less deterministic error) relative to the *Constant-Specification* case. This is a result of the the modeling issue discussed in Section 6.1.6. If the order of the function were increased, the number of future steps necessary would also increase, and the level of biasing would introduce more lagging and lack of resolution of the jumps.

Even though the two *Function-Specification* methods appear to perform adequately for the given step flux, they can not resolve oscillating fluxes as shown in Fig. 6.16. It becomes clear that the required amount of bias is much smaller than that provided by the function specification methods. However, the *Exact-Matching* method which introduces no

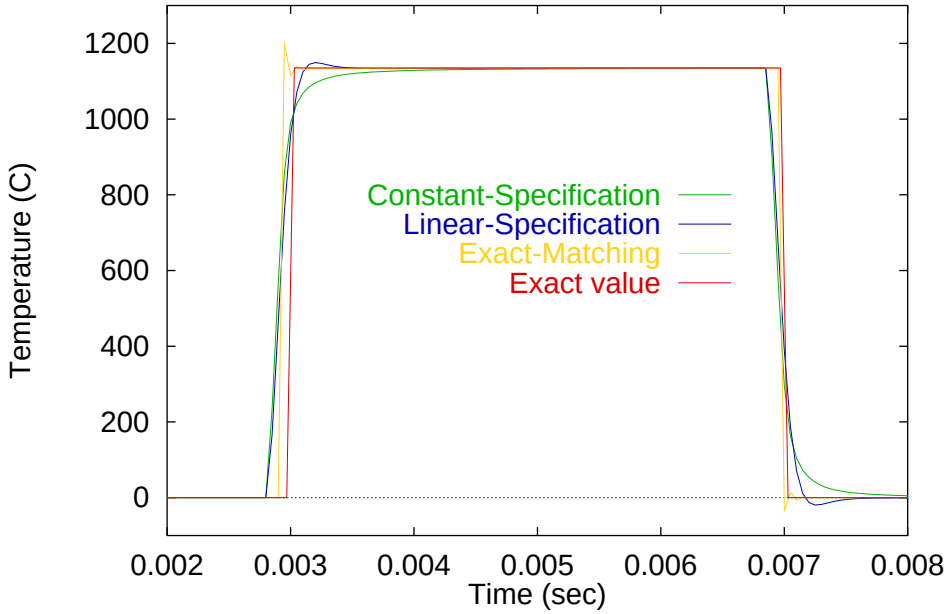

Fig. 6.15 **Increase in bias with functional order of the *Function-Specification* methods applied to the Step Flux Case.**

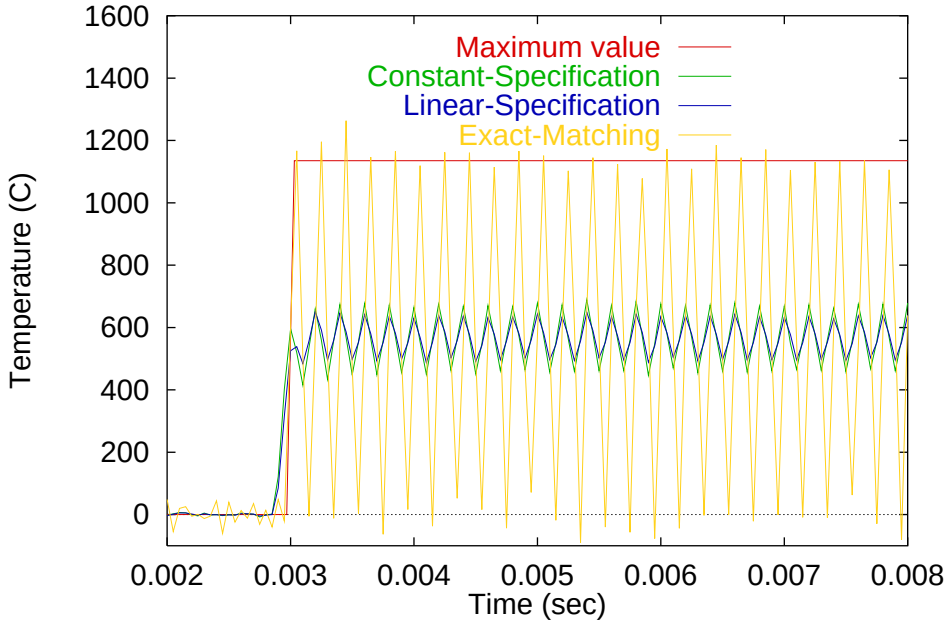

Fig. 6.16 **Lack of bias resulting from th *Exact-Matching* method compared with extreme bias of the *Constant-Specification* and *Linear-Specification* methods for the High Frequency Oscillating Test Case.**

biasing, can resolve the flux much better than its *Function-Specification* counterparts. The *Regularization* method, however, allows for infinitely variable biasing to be added to the solution.

The question arises concerning the amount of bias that we can add to the solution to mitigate the amplification of measurement errors while not introducing significant deterministic errors. The amount of biasing is controlled by the regularization parameter α and must be found through a "trial and error" approach as described in Section 4.2. If we plot the root mean squared error as a function of the regularization parameter as in Fig. 6.17, we can locate a region of low deterministic error.

Therefore from Fig. 6.17, the largest α allowable is approximately 10^{-5} for both cases. Even though we could afford more regularization in the step case, we must select the parameter conservatively. Note that a rigorous evaluation of the regularization parameter would require a similar analysis for each individual case. However, note that a minimum value of the root mean squared error can be obtained for a wide range of α. Therefore, it is safe not to perform the analysis for each test case; we simply select the worst case scenario which happens to be the oscillating flux case. In other words, we locate the smallest root mean squared error for a range of α for each test case. Table 6.5 shows the optimal regularization parameter for the different types of data.

An alternate method of evaluating the regularization parameter is to find the value that results in a root mean squared residual that is comparable to the root mean squared error of the measurement. In this manner, we have assured that the noise will not become amplified while limiting the deterministic error. This analysis yields the same results as the visual inspection of Fig. 6.17 and the evaluation of Table 6.5.

Table 6.5 Optimal regularization parameters of the *Regularization* and *Explicit-Regularization* methods for various test cases.

	α $(\mathrm{K^2m^4/W^2})$			
	Regularization		*Explicit-Regularization*	
Test Case	Noisy	Exact	Noisy	Exact
Step	5.0×10^{-5}	2.0×10^{-5}	2.0×10^{-4}	2.0×10^{-5}
High Frequency	5.0×10^{-5}	2.0×10^{-5}	5.0×10^{-5}	5.0×10^{-5}
Spike	1.0×10^{-3}	5.0×10^{-4}	2.0×10^{-3}	1.0×10^{-3}
Pulse	1.0×10^{-4}	5.0×10^{-5}	2.0×10^{-4}	1.0×10^{-4}

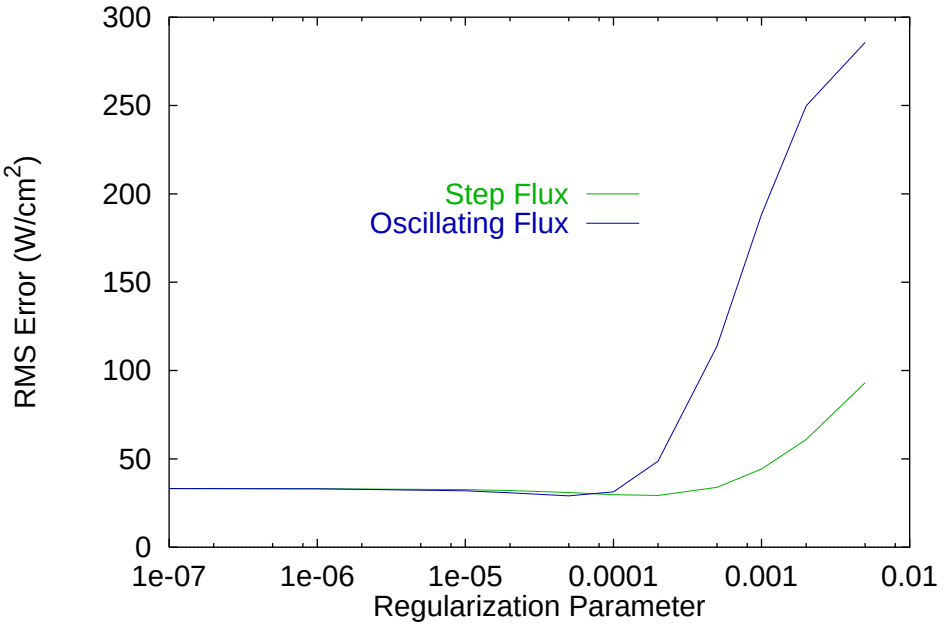

Fig. 6.17 The RMS Error as a function of the Regularization Parameter, α.

6.3 Analysis of Experimental Data

Now that we have an understanding of the methods' behavior given certain test situations, we can apply this knowledge to the analysis of real data. Since we do not know the actual flux, we can not say with certainty, which method provides the better estimate. Nevertheless, the heat flux was estimated for Run 43 and Run 85 using Class 1, Class 2 and Class 3 methods. From the estimates, a temperature response was determined with a conduction solution so that the residuals between the measured and calculated temperatures could be computed. Now, we can examine the residuals for the two sets of experimental data as in Table 6.6.

Fig. 6.18 shows the estimates from the *Explicit-Regularization* and *Finite-Difference* methods for Run 43. As expected, the inverse method results in a higher estimate at the peaks. This is because Class 1 and Class 2 methods have trouble resolving dynamic data. We must be careful to realize that while most methods amplify measurement noise, they also dampen dynamic data. These effects can offset each other to provide an estimate that appears accurate. This is the reason why several test cases were used to evaluate the performance in several situations. Ideally, the test cases will isolate particular effects so that

Table 6.6 Residuals from Run 43 and Run 85

Temperature (K)					
Method		Run 43		Run 85	
		RMS	MAR	RMS	MAR
Class 1	*Cook-Felderman*	0.62	5.63	0.17	2.08
	Kendall-Dixon	1.25	12.39	0.43	5.68
	Diller	0.71	6.56	0.23	2.99
Class 2	*Finite-Difference*	0.25	2.46	0.13	1.70
	Simple-Implicit	0.19	1.64	0.12	1.50
	Rae-Taubee	1.71	15.23	0.38	4.87
Class 3	*Constant-Specification*	0.25	2.30	0.22	2.65
	Linear-Specification	0.16	1.42	0.20	2.44
	Regularization	0.02	0.18	0.05	0.63
	Explicit-Regularization	0.03	0.29	0.04	0.46
	Exact-Matching	0.02	0.19	0.02	0.24

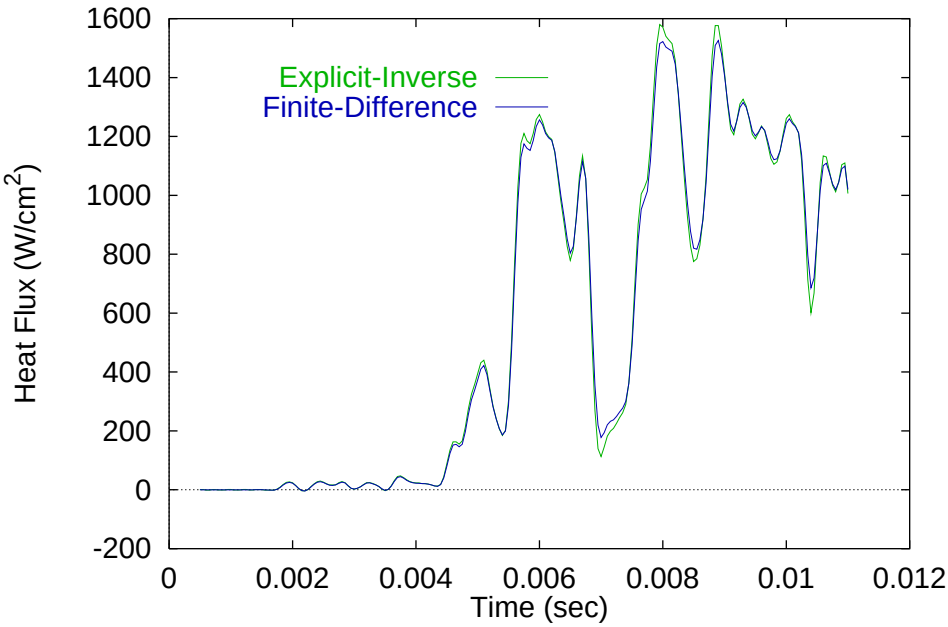

Fig. 6.18 Estimates for Run 43.

we can build confidence in certain methods while estimating real heat fluxes. Therefore, we should use the method that performs the best on all the test cases.

From the *Exact-Matching* and *Simple-Implicit* estimates for Run 85 in Fig. 6.19, we can see that the unsteady behavior is more dynamic. Also note that the *Exact-Matching* method estimates that the peaks are up to 40% larger than those of the *Simple-Implicit* method. Returning to the results for the high frequency test case, Table 6.1 indicates that the *Exact-Matching* method provides much more accurate results than the *Simple-Implicit* for this type of data. Furthermore, we can deduce the types and magnitudes of the relative estimates by examining Fig. 6.16 and Fig. 6.11. Despite noise present in the *Exact-Matching* estimate, we expect it to contain less lagging than the *Simple-Implicit* estimate.

Again, the residuals can provide insight into which methods provide legitimate estimates. Most contain some bias as illustrated in Fig. 6.20 of some of the Run 85 residuals. Note that the *Diller* method, because it lags the solution, exhibits large residuals. The inverse method (*Regularization*), on the other hand, exhibits a variance error that is on the same order of magnitude as the measurement noise. The other method displayed, the *Cook-Felderman* method, contains the same bias as the *Diller* which is due to the temperature dependent property issue. However, the variance is smaller because the *Cook-Felderman* method does not lag the measured value. It is interesting to note that the *Diller* method's global performance as given in Table 6.1 is better than the *Cook-Felderman*. However, note that the residuals given in Table 6.2 and Table 6.4 are larger for the *Diller* method over the *Cook-Felderman* method. The plot of the residuals for Run 85 also indicates more variance in the solution for the *Diller* method.

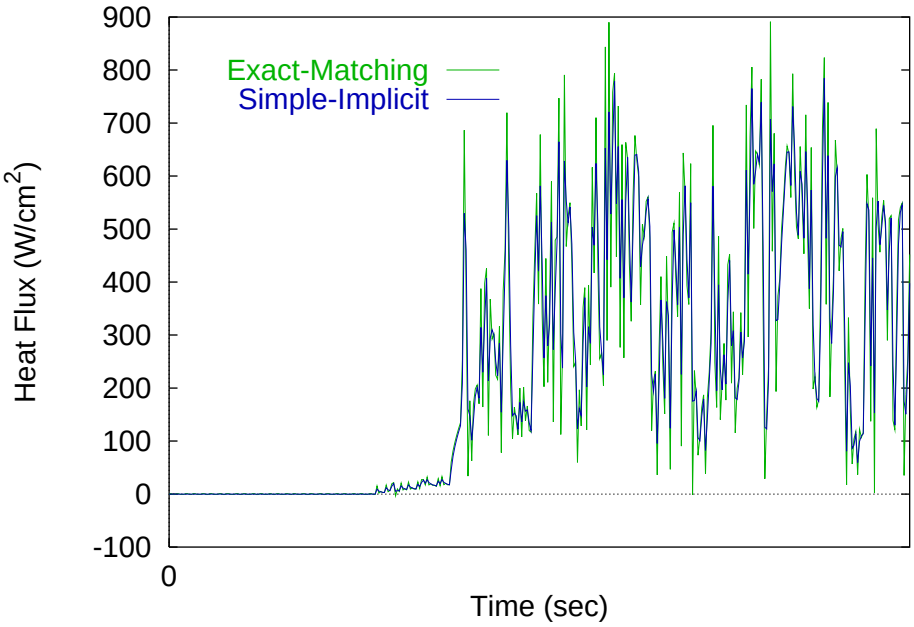

Fig. 6.19 Heat Flux estimates for Run 85.

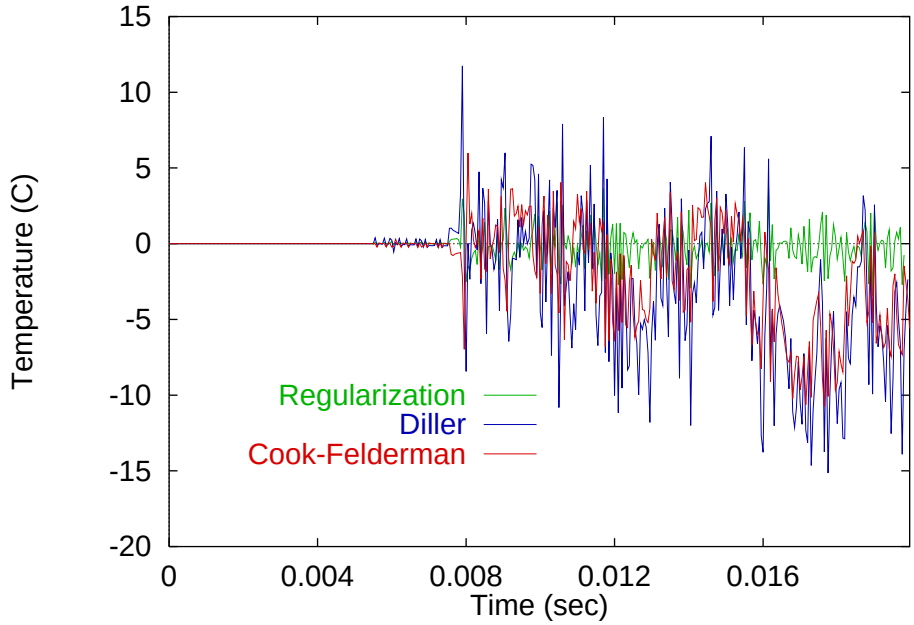

Fig. 6.20 Temperature residuals for Run 85.

Chapter 7

Nonuniformity Analysis

The second part of the analysis involves the realm of lateral conduction effects and nonuniform boundary conditions. For this analysis we must incorporate two-dimensional conduction. The methods used for the analysis and comparison were selected based primarily on the expected performance given the experimental parameters. Table 7.1 shows which method was used to obtain the estimates.

Based on the results from the one-dimensional study, the *Function-Specification* method has been selected to use as a one-dimensional estimation method to compare to the two-dimensional *Function-Specification* inverse approach when examining test cases. It was found that the parameters of the problem (and the fact that there is very little transient behavior), lend themselves to this particular estimation method. In the nonuniform study, the methods are not being scrutinized as much as the differences between the one-dimensional and two-dimensional conduction assumptions. For the experimental data, one-dimensional estimates were provided by NASA LaRC. Their method is based on a *Simple-Implicit* and will be referred to as such. Note that the *Explicit-Regularization* method was used for the two-dimensional estimation of the Upilex data. It was found that due to the new sen-

Table 7.1 One-dimensional and two-dimensional Methods used for each type of data.

Data Type	One-Dimensional Method	Two-Dimensional Method
Test Cases	*Function-Specification*	*Function-Specification*
Macor Model	*Simple-Implicit*	*Function-Specification*
Upilex Model	*Simple-Implicit*	*Explicit-Regularization*

sor spacing and surface properties, that the *Simple-Implicit* method could provide more accurate results.

Two simple test cases were designed to illustrate the existence and significance of lateral conduction effects. The selected methods will be used to estimate fluxes from the test data. Then experimental data will be examined by the same two methods.

7.1 Analysis of Test Cases

The development of a new method to resolve spatially dependent heating rates is grounded in the application of needing to resolve the localized heating rate produced by hypersonic jets in shock-shock experiments. Therefore, the example problem identified here is modeled after an experiment run at NASA Langley's 20" Mach 8 wind tunnel. The experiment and the data are described by Berry and Nowak (1996).

Since we are only interested in the "edge" of a single step, we have reduced the problem to seven sensors and twenty time steps that include this feature. The hypothetical experiment will run for 0.4 sec with an applied flux ramping in time from zero at $t_o = 0$ sec to 100 W/cm^2 at $t_f = 0.4$ sec. Note that the flux is only applied to a portion of the test region. The remaining area will receive zero flux. Two cases will be examined where the location of the step is directly on a sensor (Test Case 1) so that one half of the surface will be heated and where the location of the step is midway between two sensors (Test Case 2) that lie in the middle of our domain as displayed in Fig. 7.1.

The conduction model is a 1.3 cm diameter Macor cylinder with temperature dependent properties. (Macor is a machinable ceramic.) The sensors are placed 0.0635 cm apart so the length of our test section is only 0.38 cm long.

Two two-dimensional temperature distributions resulting from each of the two test cases described above were determined using an independent conduction solution. The forward conduction solution was obtained using EAL, a finite element package. The FE script is given in Appendix E and the code verification is given in Appendix C. These temperature responses were then used as the test data from which flux estimates were obtained using the one-dimensional and two-dimensional approaches. The root mean squared errors of the estimates are displayed quantitatively in Table 7.2. Note that the errors are relative to a flux of the heated area which ramps from zero to 100 W/cm^2. Since the errors of the esti-

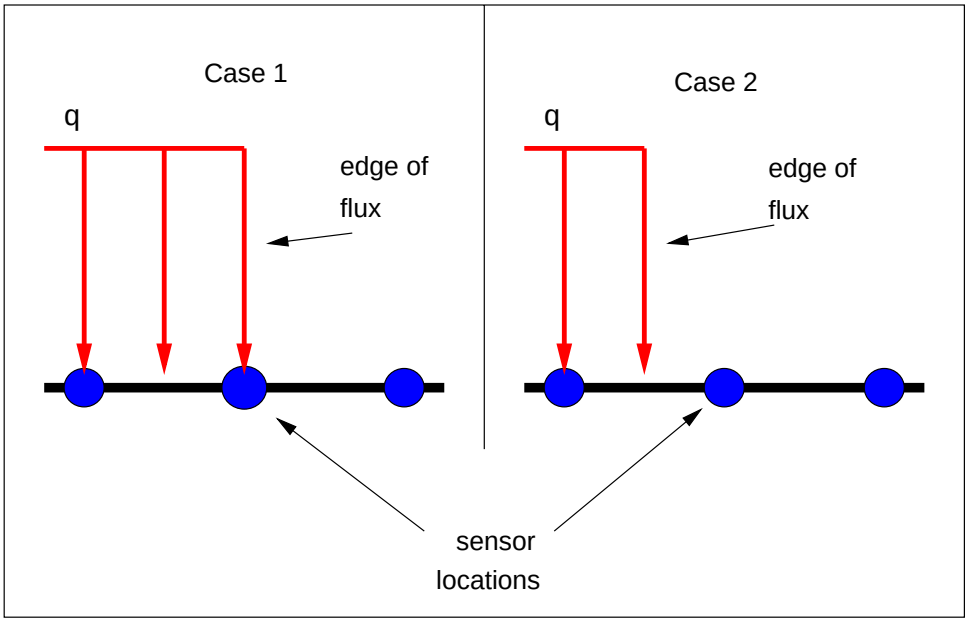

Fig. 7.1 The difference in the location of the applied spatial step flux between the two test cases.

Table 7.2 Error in the flux estimate of the two test cases for the one-dimensional and two-dimensional methods.

Root Mean Squared Error(W/cm^2)		
Method	Case 1	Case 2
Two D *Function-Specification*	0.485	0.204
One D *Function-Specification*	3.253	3.305

mates are much smaller for the two-dimensional approach, it is immediately apparent that conduction effects are significant in determining the heat flux from the surface temperature measurements. Note that for Case 1, where the location of the calculated flux coincides with the singularity, an average value of $50\,\mathrm{W/cm^2}$ was used as the flux at this point for root mean squared error calculation purposes.

A better understanding of the strengths and weaknesses of the two approaches can be obtained with a qualitative analysis. Therefore, outlined below are the results of the estimation from a one-dimensional inverse technique and the two-dimensional inverse technique described earlier. The applied heat flux for Test Case 1 is shown in Fig. 7.2 and the resulting temperature distribution for the forward problem is shown in Fig. 7.3. For test case 2, we simply shifted the location of the step so that it occured between two sensors instead of on the sensor (see Fig. 7.1). The temperature distribution should be identical for both cases in a continuous sense. When the continuous distribution is discretized, the distribution will be slightly different. Herein lies the difficulty in evaluating nonuniform heating rates.

The flux estimates of the test cases for the one-dimensional approach and the two-dimensional approach appear different as seen in Fig. 7.4 compared to Fig. 7.5 where the entire space for Test Case 1 is vizualized and in Fig. 7.6 where the estimates for Test Case 2 at the final time are compared. Remember these figures depict the estimations' attempt to recover the exact flux as in Fig. 7.2. Immediately, we notice that the two-dimensional estimate resolves the step with greater accuracy especially for Test Case 2, where the step is placed between sensors. As expected, both estimation routines have trouble when the step occurs directly on a sensor. However, the two-dimensional approach resolves (more accurately) a steeper gradient at the step location. Note the slight instability on either side of the step. To determine which estimate is actually the more accurate, refer back to Table 7.2 which gives the root mean squared error for each estimate shown for the test cases. Based on these findings for the performance of the estimation techniques, we can apply the methodologies to actual experimental data.

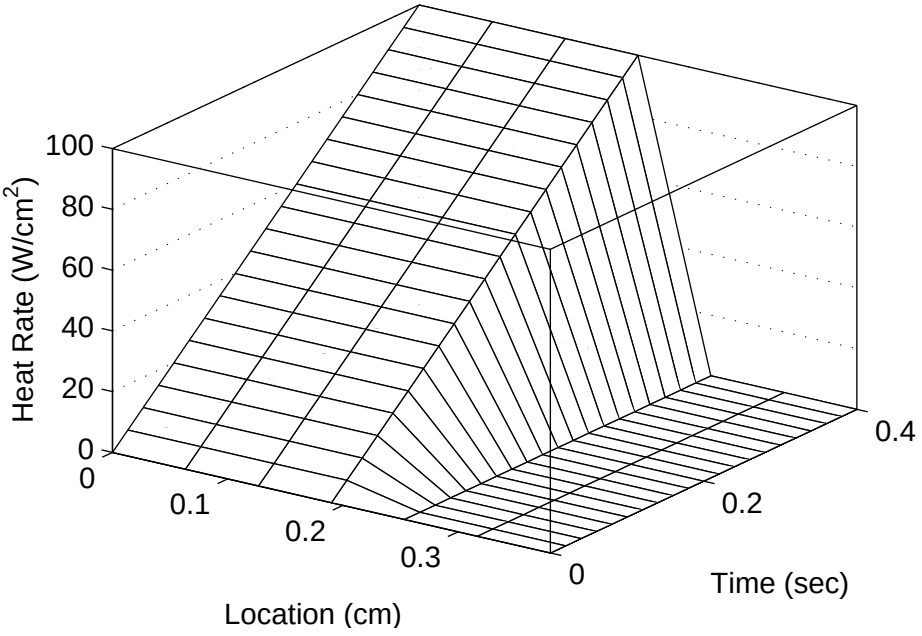

Fig. 7.2 Exact flux of Test Case 1.

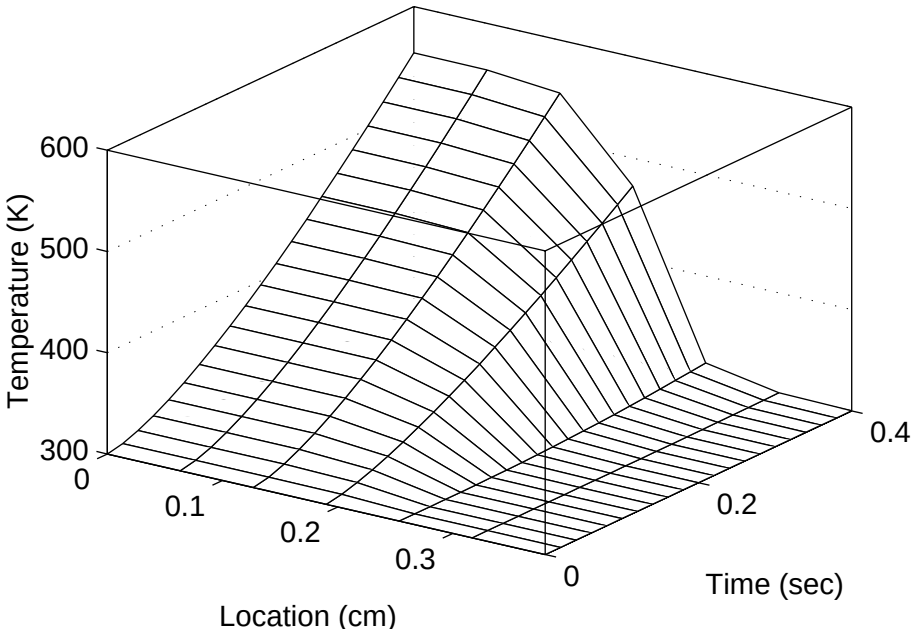

Fig. 7.3 Surface temperature response to the step in space of Test Case 1 used as temperature measurements.

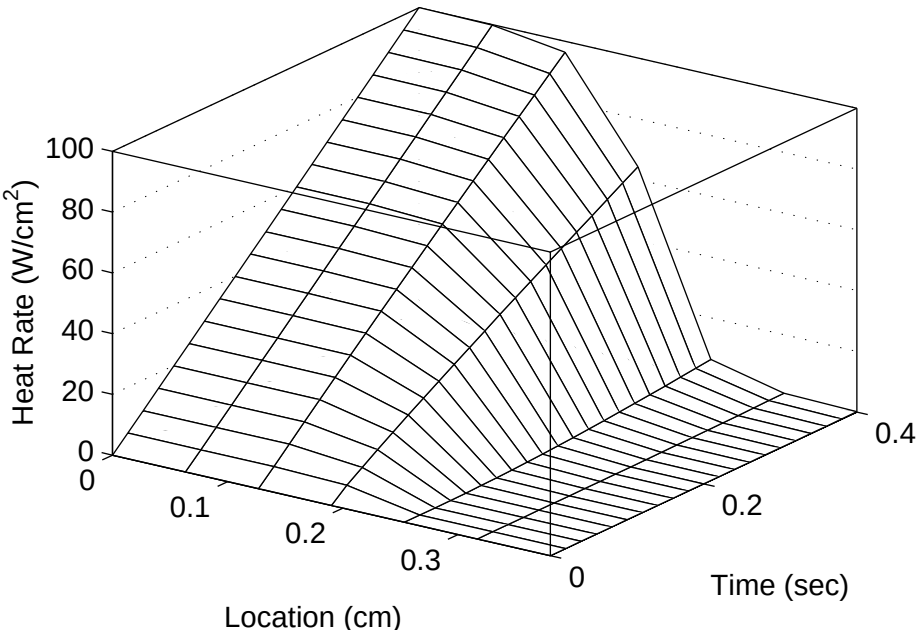

Fig. 7.4 One-dimensional estimate of Test Case 1.

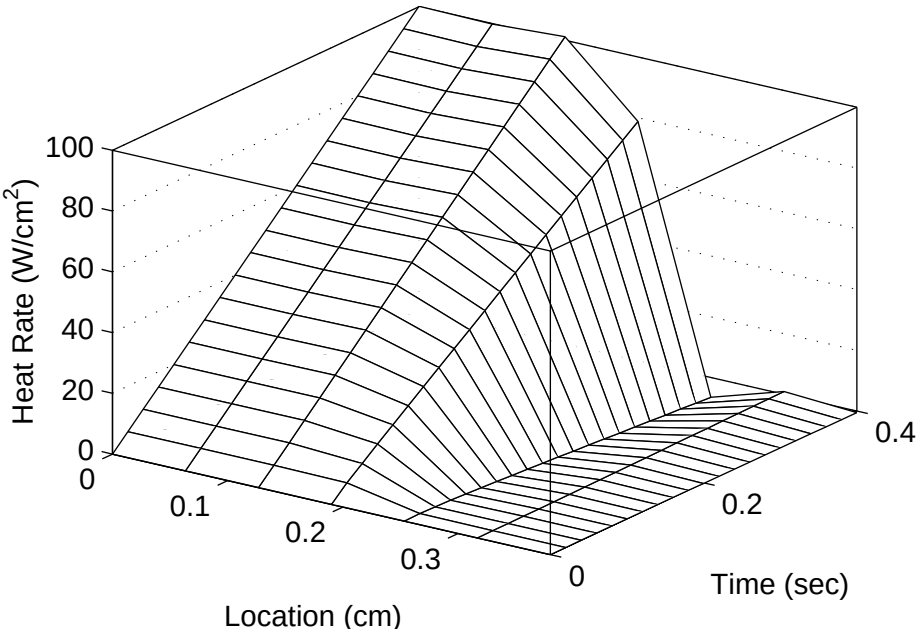

Fig. 7.5 Two-dimensional estimate of Test Case 1.

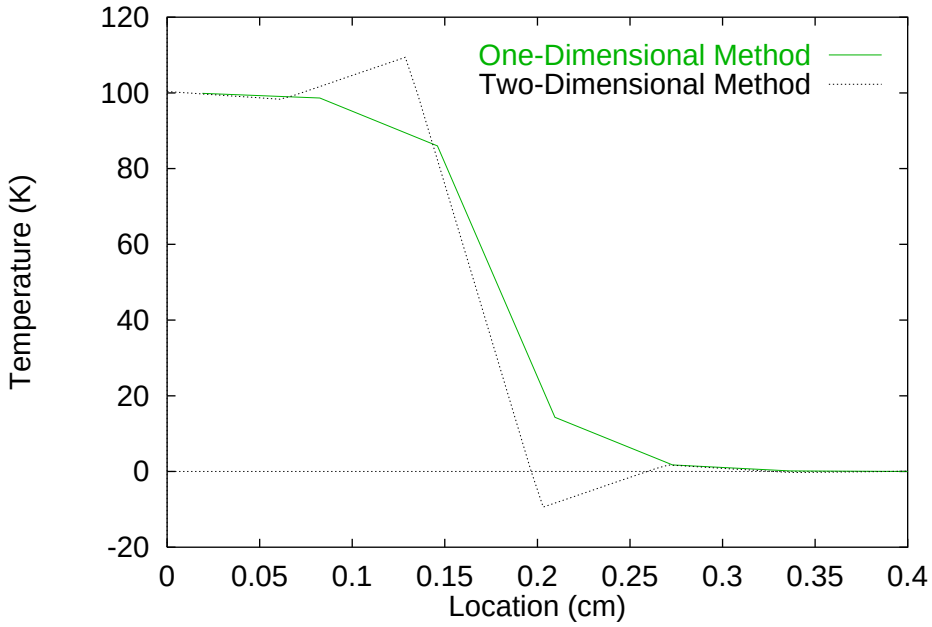

Fig. 7.6 One-dimensional and two-dimensional estimates of Test Case 2 at the final time ($t = 0.4\,\text{sec}$).

7.2 Analysis of Experimental Data

A schlieren of the shock interaction and resulting jet impingement for Run 14 is shown in Fig. 7.7. Note the interaction is three-dimensional in nature, so the gossamer edges are a result of the shock projected in two dimensions. As a result, the flow arrows are only approximations to the actual flow patterns. The length of the cylinder that is displayed in the image covers approximately 2 cm in the region of peak heating. In this region, the sensor spacing is either 0.038 cm or 0.0635 cm depending on the test model used (see Fig. 2.6 for details). A rough measurement reveals that the width of the jet is barely two sensors wide at best. Herein lies the difficulty of capturing the peak heating rate and location of the impingement.

7.2.1 Macor Model

The temperature response for Run 14 is given in Fig. 7.8 where the jet impingement can be identified as the location of greatest temperature. "Below" the peak (to the right on the graph) is the quiet region where the heating is a result of stagnation heating. Immediately after the startup period (where nothing appears to happen), the temperature rises suddenly

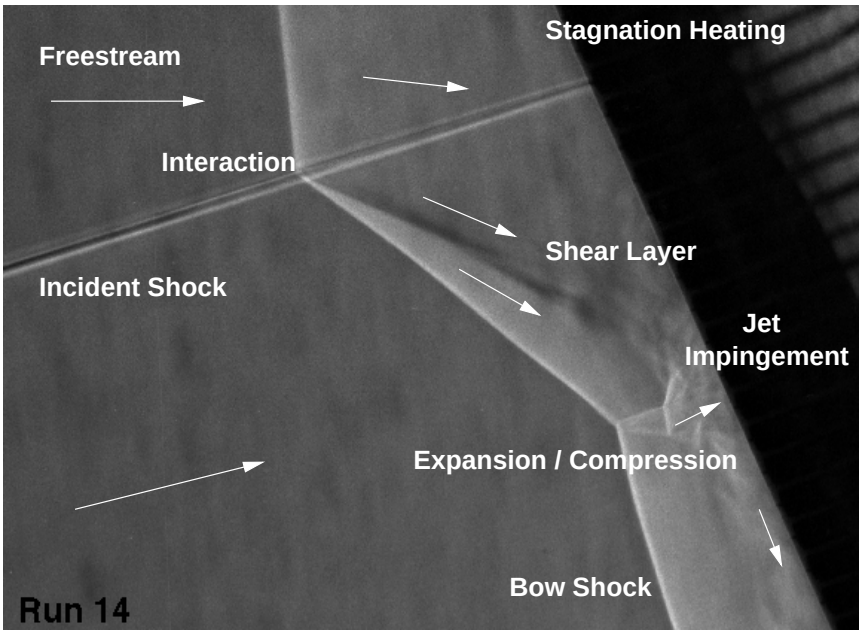

Fig. 7.7 **The two-dimensional Schlieren projection of the three-dimensional interaction of Run 14. The arrows indicate flow direction.**

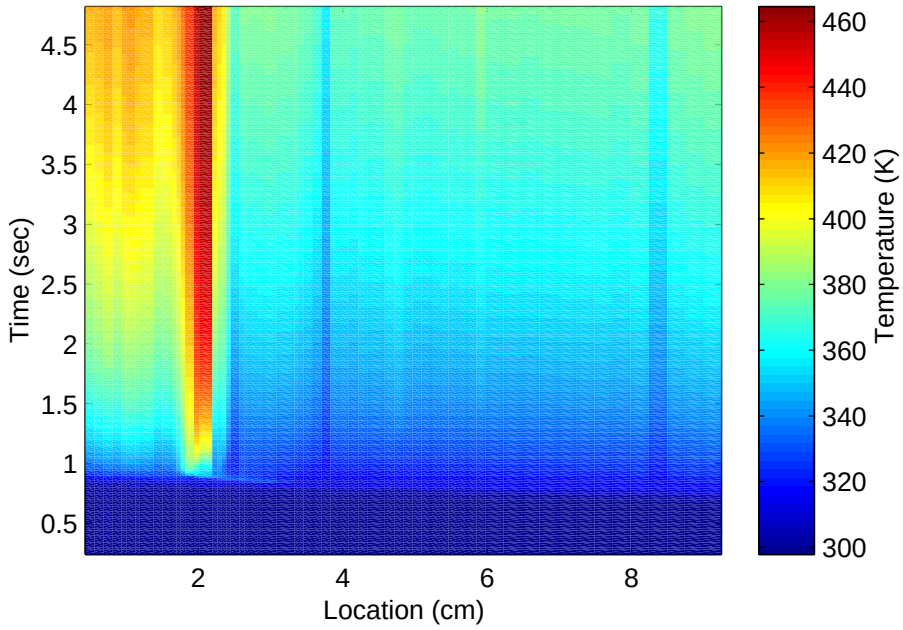

Fig. 7.8 **Measured Temperatures for Run 14**

which will correspond to an abrupt increase in the heat flux. The temperature then continues to rise as the heat flux becomes more steady. Note that the region away from the peak heating contains very little features, in other words, it is being heated as if it were purely stagnation heating, and there will be very little lateral conduction effects. Finally, notice the secondary heating locations flanking the primary heating location. These are due to expansion and compression waves as described in Section 2.1.

Two estimates for the heat flux were obtained for Run 14 temperature measurements. The first (assuming one-dimensional conduction), was generated by code developed at NASA LaRC (Hollis, 1995) and is most similar to the *Simple-Implicit* method described here. The second method is a *Function-Specification* method which assumes two-dimensional conduction. For stability and accuracy, it was found that the 3-node triangular patch with the *Linear-Specification* approach provided the best estimates. A finite element solution method along with the inverse technology was employed via a scripting language called EAL (Whetstone, 1983) to calculate the estimate.

The two estimates are shown in Fig. 7.9 and Fig. 7.10 for the two-dimensional and one-dimensional approaches respectively. As expected, the two-dimensional results contain higher and narrower peaks. Furthermore, the secondary heating, located adjacent to the main heating location, is more evident with the two-dimensional analysis. The darker stripes located in the quiet regions are sensors that failed during the experiment and do not represent a flow phenomenon. Note that at later times, after much of the energy has diffused through the model, the one-dimensional estimate also smears the estimate. More closely matching physics, we notice that the two-dimensional estimate remains steady during the experiment as expected.

The residuals of the estimates can lend a quantitative insight into the performance of the methods. Recall that the residual represents some difference of temperatures given by Eq. (5.2). Table 7.3 shows the enhanced performance of the two-dimensional method. The temperature response of each estimate was calculated using a multi-dimensional forward conduction solution. This response was then compared to the measured temperatures and plotted in Fig. 7.11. The comparison of the residuals indicates that the one-dimensional method can not resolve the peak as well as the two-dimensional method simply because the magnitude of the residuals is larger. Further note that the one-dimensional approach has alternating regions of positive and negative residuals within the peak heating region. This is

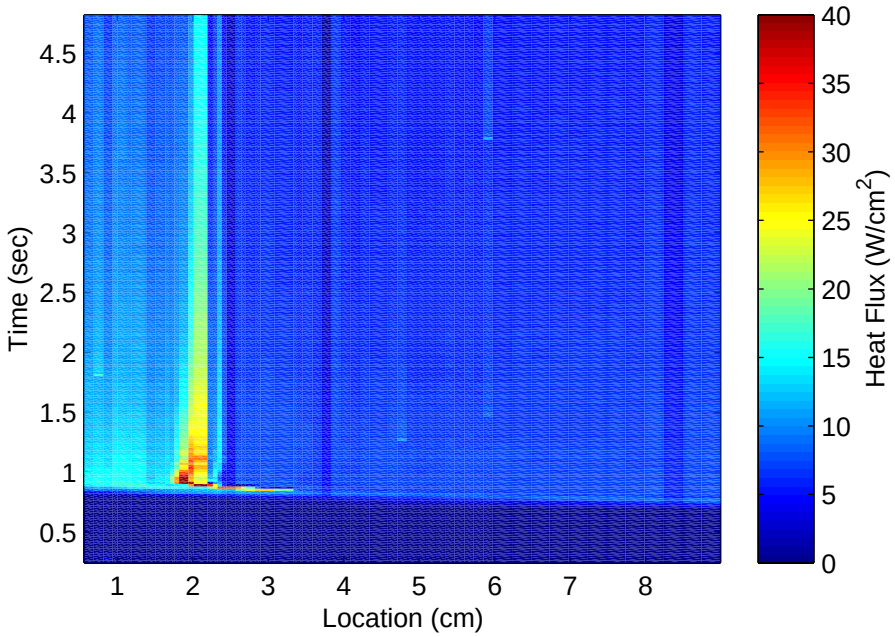

Fig. 7.9 Two-dimensional heat flux estimates for Run 14.

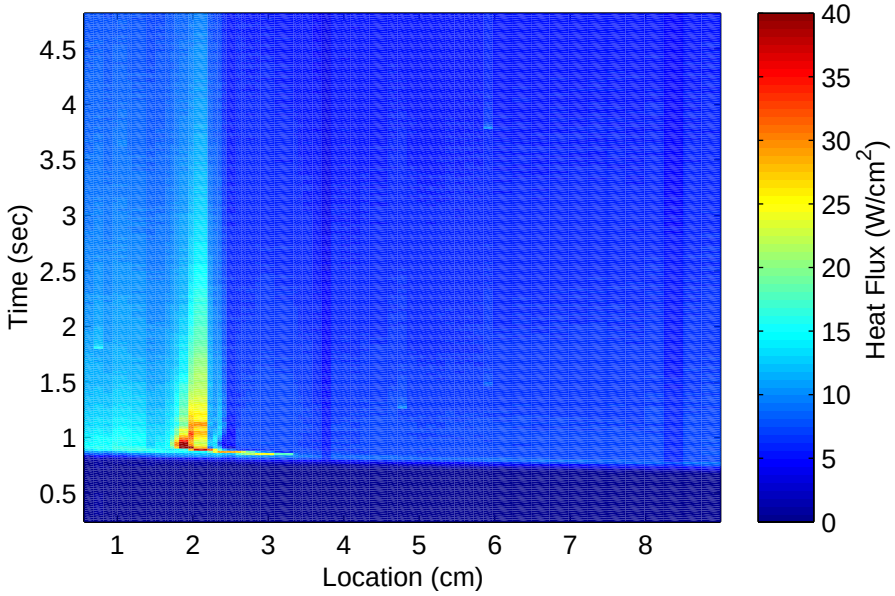

Fig. 7.10 One-dimensional heat flux estimates for Run 14.

Table 7.3 Temperature Residuals for experimental data.

Temperature (K)				
	1D		2D	
	RMS	MAE	RMS	MAE
Run 14	0.055	2.300	0.019	0.920
Run 36	0.055	2.138	0.019	0.715
Run 58	0.123	8.298	0.029	1.557
Run 60	0.127	8.401	0.041	1.865

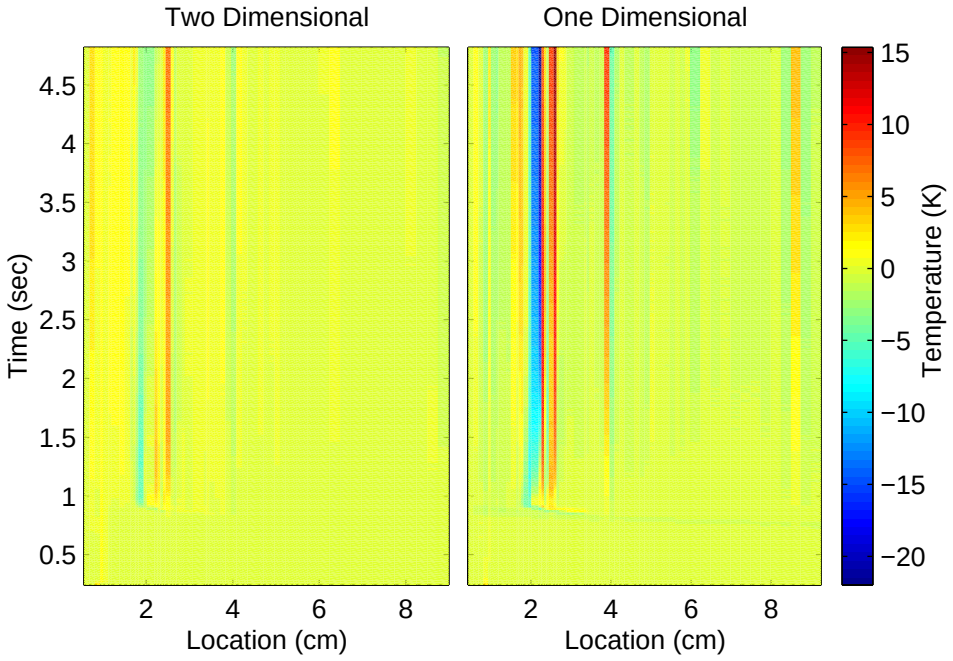

Fig. 7.11 Temperature residuals of estimates of Run 14.

a result of the one-dimensional method "smearing" the estimate because lateral conduction effects can not be accounted for. This effect can be seen more clearly by examining the estimate at a single time as in Fig. 7.12.

At two seconds, the flow has stabilized so we can concentrate on the "shape" of the flux instead of the transient features, and this is the time at which the transients are assumed to be quiet by the original experimenters (Berry and Nowak, 1996). The one-dimensional estimate at this time (and for all times) does not resolve the peaks and valleys as pronounced as the two-dimensional estimate. Again, the two-dimensional estimate is thought to be more accurate because the residuals are smaller. Analysis of the test cases also confirms the validity of the two-dimensional estimate. As an additional check, notice that the residuals for both methods (in Fig. 7.11) are relatively small in the quiet region, and that the estimates are almost the same in this region (Fig. 7.12). Where there are no spatial gradients of temperature, we expect the two-dimensional method to behave as the one-dimensional method. Since there are no spatial gradients in the quiet region, there are no conduction effects, and the estimates should be the same.

Using the same test model and experimental configuration, Run 36 provides additional data for comparison of the two approaches. The results are similar and a repetition of the discussion is unnecessary here. To demonstrate the repeatability of the two-dimensional estimation procedure (and for completeness), the measured temperature data are shown in Fig. 7.13. The one-dimensional and two-dimensional estimates are given in Fig. 7.16 and 7.15 respectively. Finally, the residuals are are presented in Fig. 7.14.

The reader will notice that the color scheme used for the residuals appears to change. However, note carefully the scale on the residual plots; the zero point is not always in the center of the scale. The feature to look for is the uniformity of the color. If it is uniform, the residuals are all nearly zero. Where there are large color differences, the residuals are large and change frequently.

To obtain an idea about the relative magnitude of the peak heating compared to the quiet region, Fig. 7.17 gives a representation of the surface created by the two-dimensional estimate. Note that only the region of interest has been displayed. To include the entire data set renders the graph indecipherable. Also realize that this is the same data as were presented in Fig. 7.15; the plot style has been changed to reveal the relative magnitude of the peak heating region. Compare the one-dimensional estimate in Fig. 7.18 to the

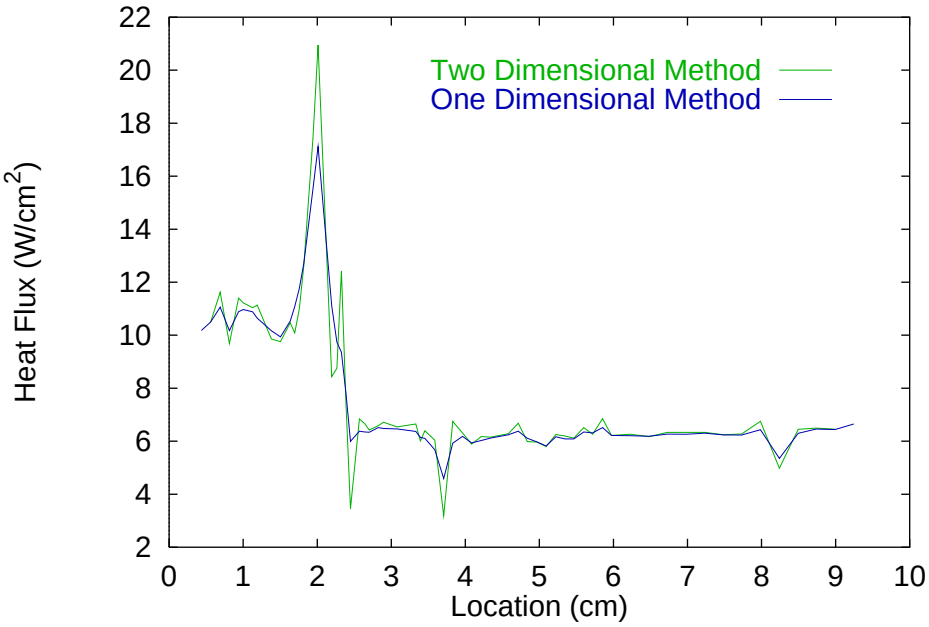

Fig. 7.12 Heat Flux estimates for Run 14 at 2 sec.

two-dimensional estimate in Fig. 7.18. Notice how the one-dimensional estimate causes the peak to steadily decrease while the physics of the problem suggest that it should remian constant. Additionally, the one-dimensional estimate can not resolve the secondary heating region at all.

Concerning this view of Run 36, the spikes and ridges that appear in the quiet region have been attributed to data acquisition noise or failing sensors (Berry and Nowak, 1996). We can also notice how this data set is not ideal for studying lateral conduction effects because of failed sensors in the nonuniform region. Each line in the plot represents a sensor location; many sensors have failed and are conspicuously missing from the analysis. Nevertheless, realize that the relative magnitude of the peak heating region is at least four times that in the quiet region and is at least ten times as much at the start of the experiment. Another apparent feature from this type of presentation is the fact that the heating rate is higher on one side of the impingement than the other. This is the region that experiences the stagnation heating behind the bow shock that develops in the free stream (region ② in Fig. 2.2). The other side (region ⑤) experiences strong compression and expansion waves near the point of impingement and primarily stagnation heating (at a lower Mach number than the other region) further from the impingement location.

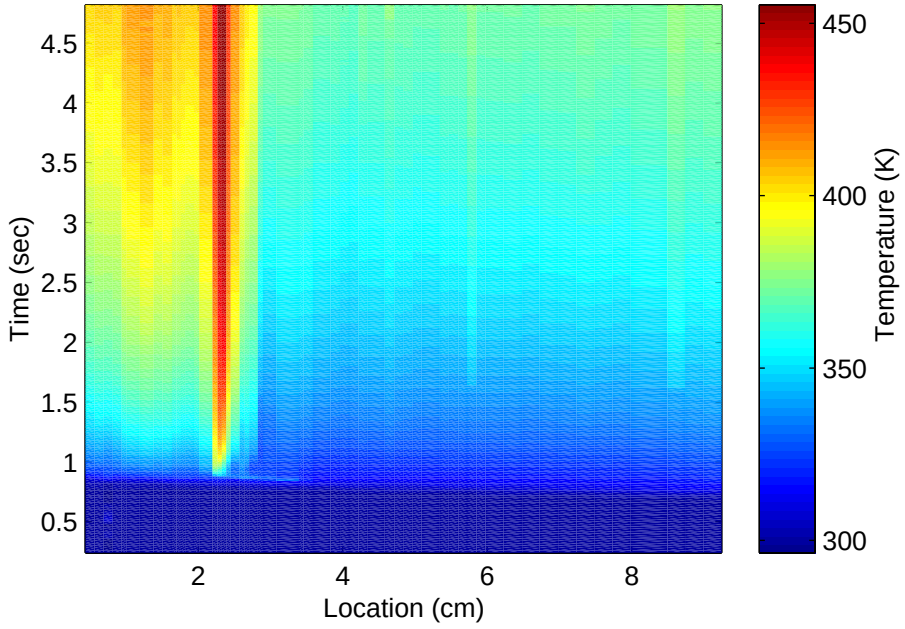

Fig. 7.13 Temperature measurements for Run 36.

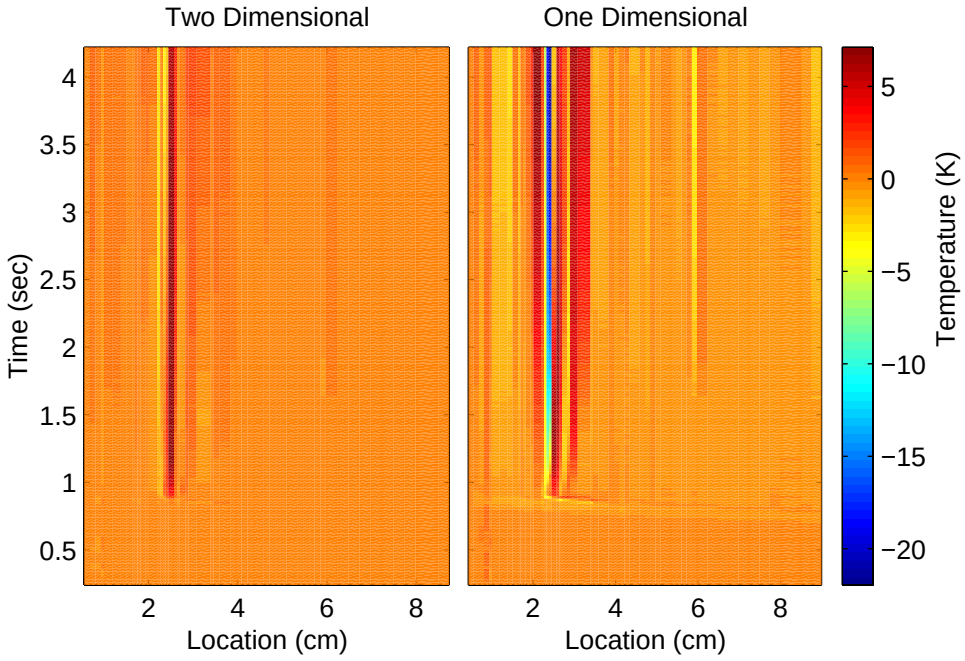

Fig. 7.14 Temperature residuals for Run 36.

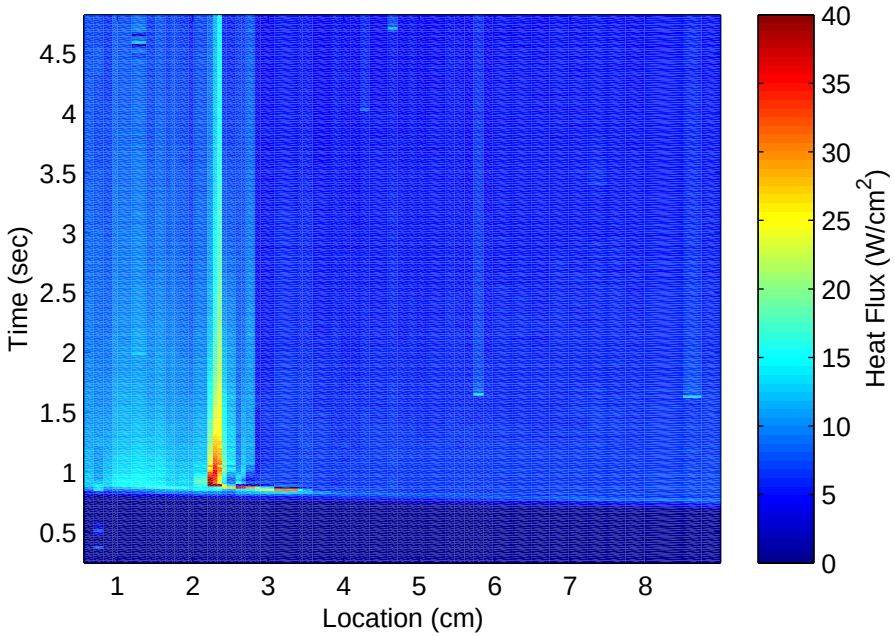

Fig. 7.15 Two-dimensional heat flux estimates for Run 36.

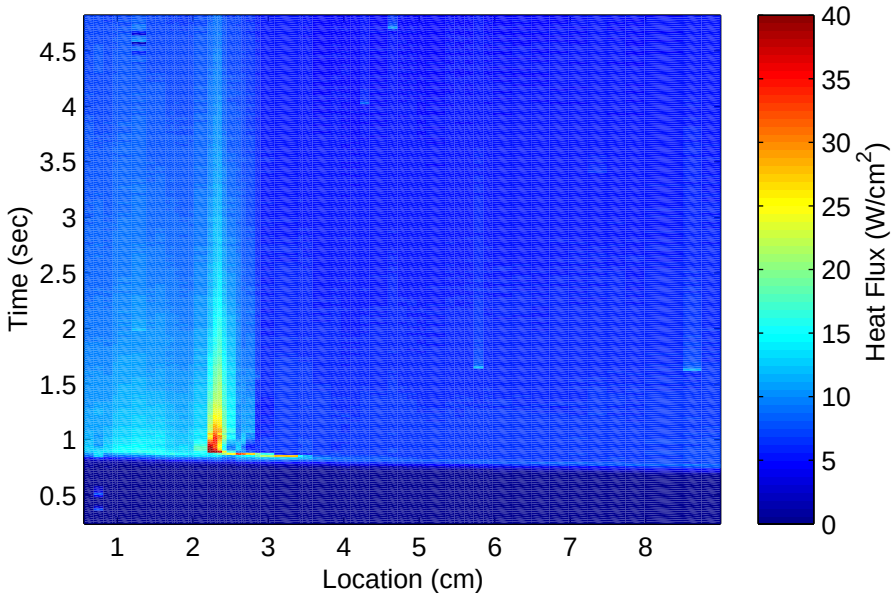

Fig. 7.16 One-dimensional heat flux estimates for Run 36.

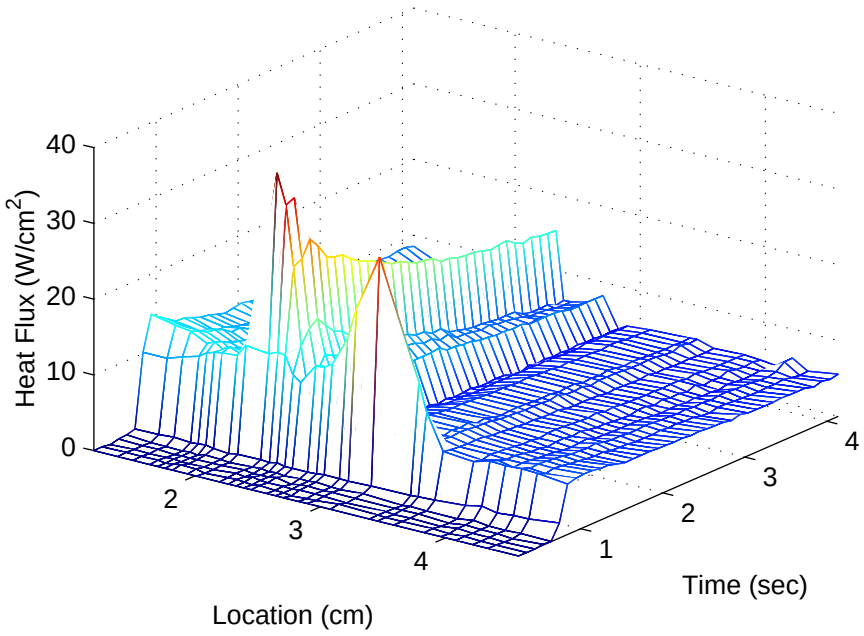

Fig. 7.17 Two-dimensional flux estimates for Run 36.

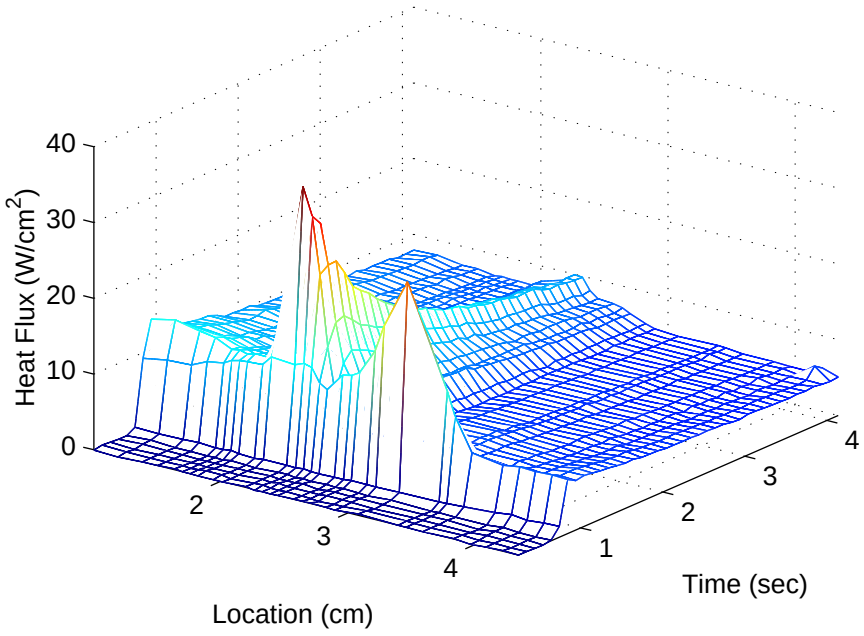

Fig. 7.18 One-dimensional flux estimates for Run 36.

7.2.2 Upilex Model

The remaining sets of data that are being analyzed make use of the Upilex model which was built with a higher sensor density in the impingement region. The Upilex model introduces additional complexity into the problem. We must consider conduction through three material layers with significantly different thermal characteristics. The sensitivity of the problem will be changed as a result of the new conduction model. The sensitivity is also changed as a result of the sensor spacing. Remember the sensitivity is defined as the derivative of the temperature with respect to the surface flux. When lateral conduction effects are included, the properties between sensors as well as the relative location of the sensors to each other affect how much diffusing energy is seen by neighboring sensors.

As a result of the new sensitivities, it was found that the *Explicit-Regularization* method could provide better estimates than the *Function-Specification* method. This decision was also partially driven by the fact that the *Explicit-Regularization* method is computationally cheaper. Because of the higher data density, the computational times using the function specification method resulted in the methodology becoming prohibitively expensive. For the *Explicit-Regularization* method, the regularization parameter had to be determined to obtain reasonable results. The two methods described in Section 6.2 were used to make this determination. It was found that a value of $\alpha = 0.001\,\mathrm{K^2m^4/W^2}$ was optimal.

For the analysis of the Upilex model data, we have focused on the region that contains the largest temperature gradients in the spatial direction. In this way, the effects of lateral conduction can be isolated and studied more closely. The first set of data is from Run 60 and the measured temperatures in Fig. 7.19 look somewhat similar to those of Run 14; the flow patterns should be identical. Because the flow structures are theoretically the same, we expect the incident heating rates to be theoretically identical as well. However, the temperature signature of the two runs should be different because of the different conduction properties between the Macor and Upilex models. Therefore, if the estimates for both the Upilex and Macor models are similar in magnitude with a similar peak width, we will have proved that the conduction model is accurate, the physics of the problem have been adequately accounted for, and the repeatability of the experiments has been verified.

Outside the peak heating region the sensor density becomes lower. This feature

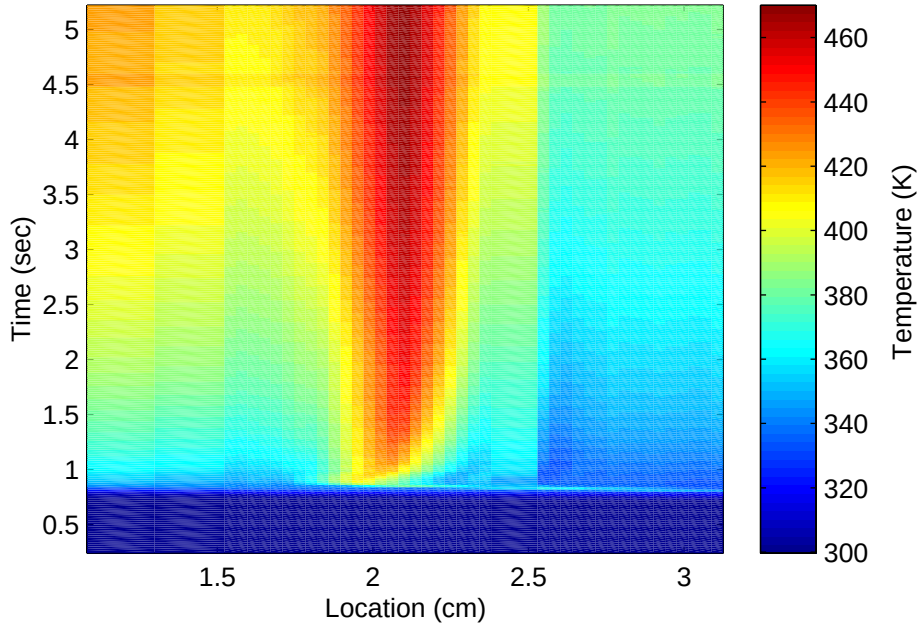

Fig. 7.19 Measured temperatures for Run 60.

shows up in a visualization of the data (such as Fig. 7.19 as regions of "constant" temperature. Actually this is an artifact of the plotting method. Since the data is discrete, the plotting method must make some assumption about the change in temperature from one sensor to the next. I have selected a constant assumption so that regions of low sensor density and regions of failed sensors can be readily seen. Furthermore, this representation of the data does not degrade the significance of the findings.

Through visualization of the temperature response (Fig. 7.19), we notice that there is a section adjacent to the peak heating region that contains low sensor density data. The sensors in this region were damaged at some point early in the experiment, and the data are, therefore, unavailable. Keep in mind that this region is corresponds to the secondary heating location. Due to the low resolution we do not expect to be able to reach definitive conclusions about the ability of either method to resolve peaks and/or valleys in this region.

The two-dimensional and one-dimensional estimates for Run 60 are displayed in Fig. 7.20 and Fig. 7.21, respectively. Again the peaks are higher and the valleys deeper in the two-dimensional estimate as opposed to the one-dimensional estimate as described for Run 14. Also the estimate of the peak heating remains intact through out the experiment for the two-dimensional method whereas it tends to fade for the one-dimensional method.

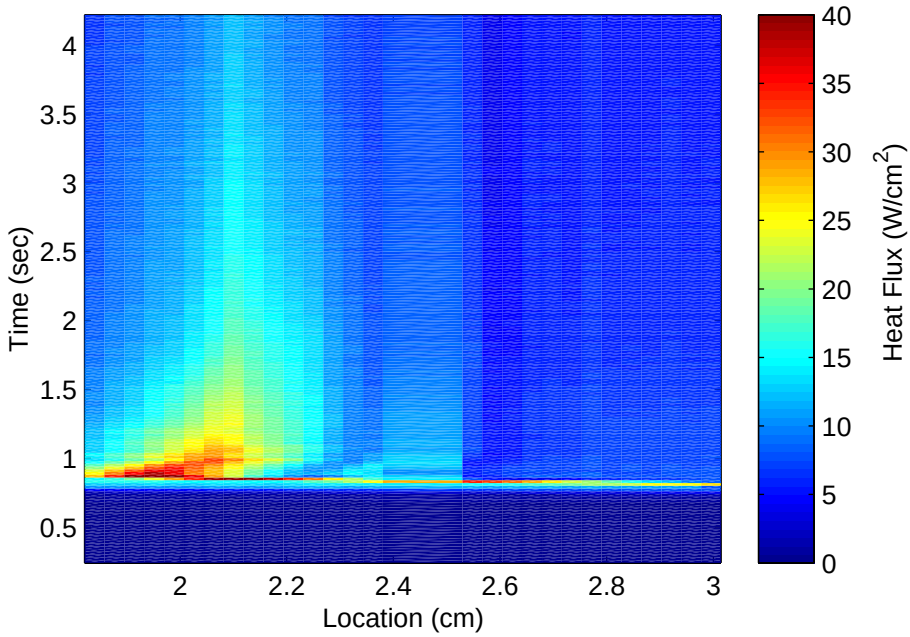

Fig. 7.20 Two-dimensional heat flux estimates for Run 60.

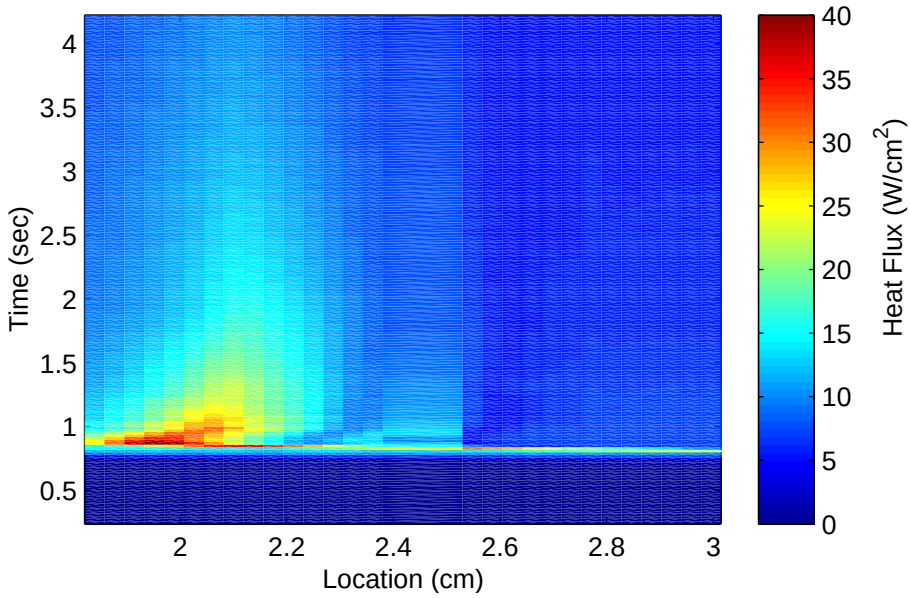

Fig. 7.21 One-dimensional heat Flux estimates for Run 60.

Despite the lack of information in the secondary heating region, notice that there is evidence of a secondary heating location in the two-dimensional estimate, while the one-dimensional approach nearly eliminates any evidence of a peak in this region.

Using the same Upilex model, data from Run 58 should be similar to that of Run 60. We should, therefore, be able to examine the region of secondary heating. Even though these data provide the information in the secondary heating region, the sensors located in the region of peak heating were damaged for this run. Therefore, Run 58 was not used for the primary analysis of the two-dimensional problem. The results, however, are reported here for completeness. The measured temperature is shown in Fig. 7.22, with the estimates for the one-dimensional and two-dimensional methods respectively in Fig. 7.25 and Fig. 7.24. From these graphs we can make out the secondary peak adjacent to the primary peak. Notice that the one-dimensional estimate does not resolve this feature at all. The residuals are visualized in Fig. 7.23. Notice the missing data in the results and the poor resolution of the secondary heating by the one-dimensional method.

7.3 Comparison to Previously Published Results

When studying shock interactions, the peak heating rate is often of interest. Therefore, the difference between the two-dimensional and one-dimensional approaches in the region of peak heating as shown in Fig. 7.12 warrants a closer look. We expect the estimates for the four runs to be similar, since the experimental flow conditions are theoretically identical. This means that the heat flux is independent of material properties and sensor construction. Note that the pictorial representation of the heat flux at 2 seconds (Fig. 7.26 for the Macor model and Fig. 7.27 for the Upilex model) indicates a difference between the models.

This difference is probably due to the fact that the properties of Upilex and the exact construction are not well known. Note that the location of the peak also changes with the experiment. This was expected due to slight modifications in the experiments. For this analysis it allows us to view each case separately while qualitatively comparing peak values. More importantly, however, note the difference in the peak heating between the one and two-dimensional approaches; we can immediately see an increase in the peak load for the two-dimensional case. Berry and Nowak (1997) suggested that this peak should be 30–40% greater once conduction effects are accounted for. The increases reported

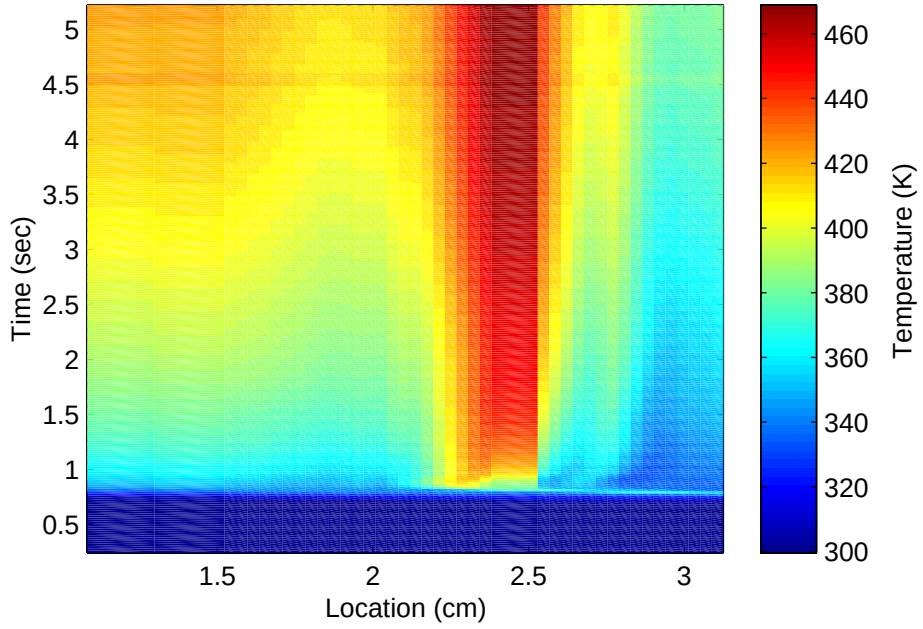

Fig. 7.22 Temperature measurements for Run 58.

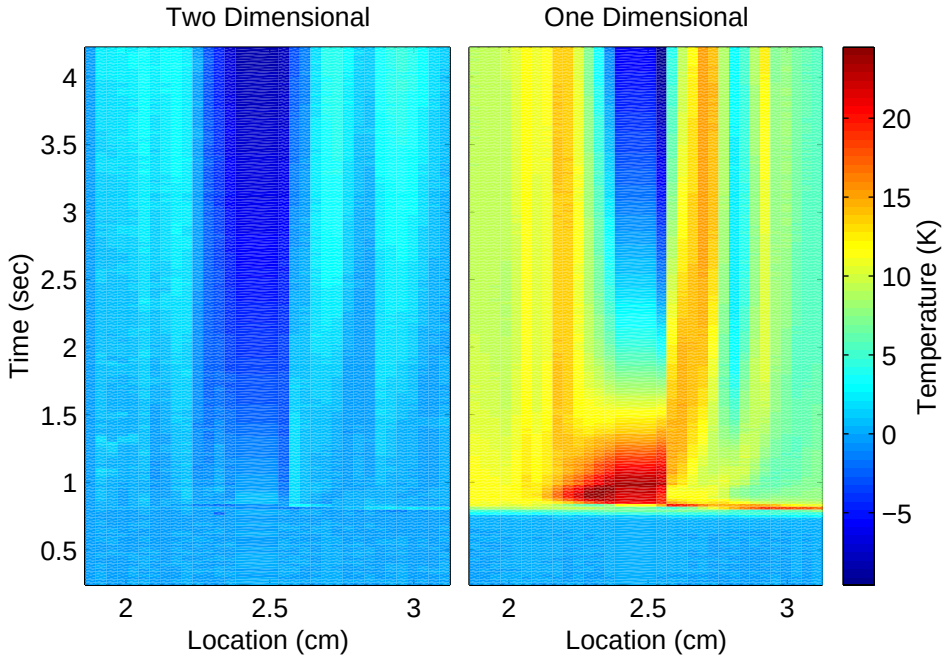

Fig. 7.23 Temperature residuals for Run 58.

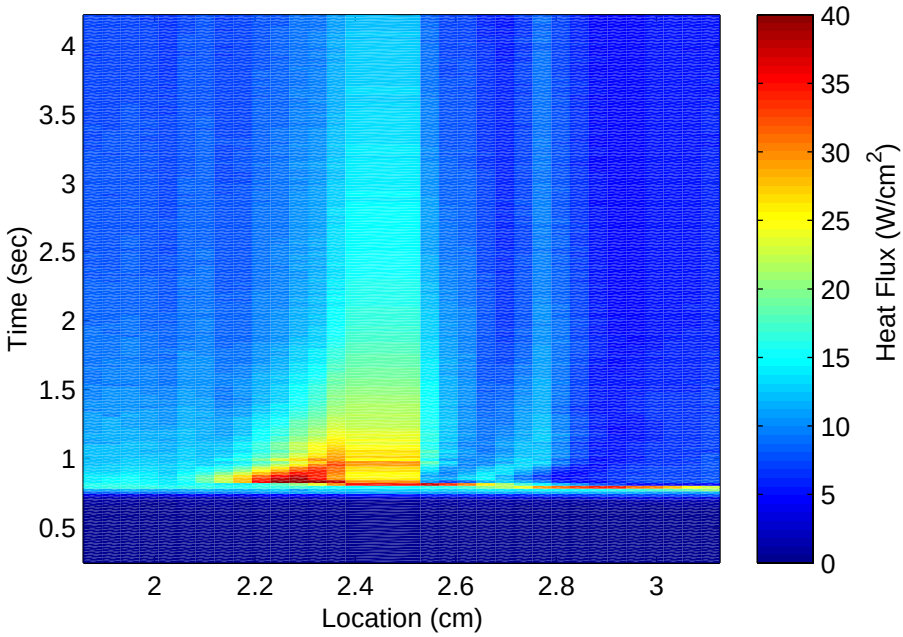

Fig. 7.24 Two-dimensional heat flux estimates for Run 58.

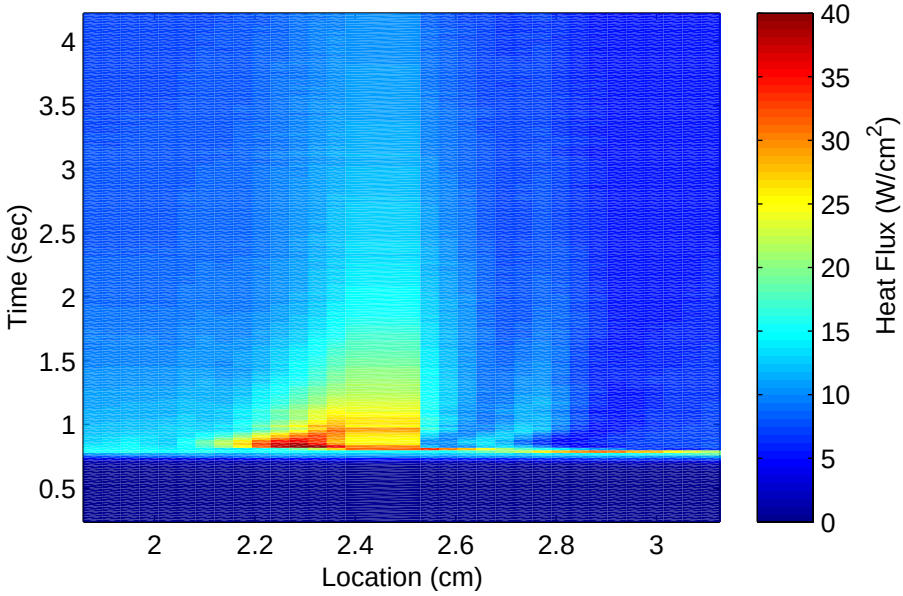

Fig. 7.25 One-dimensional heat flux estimates for Run 58.

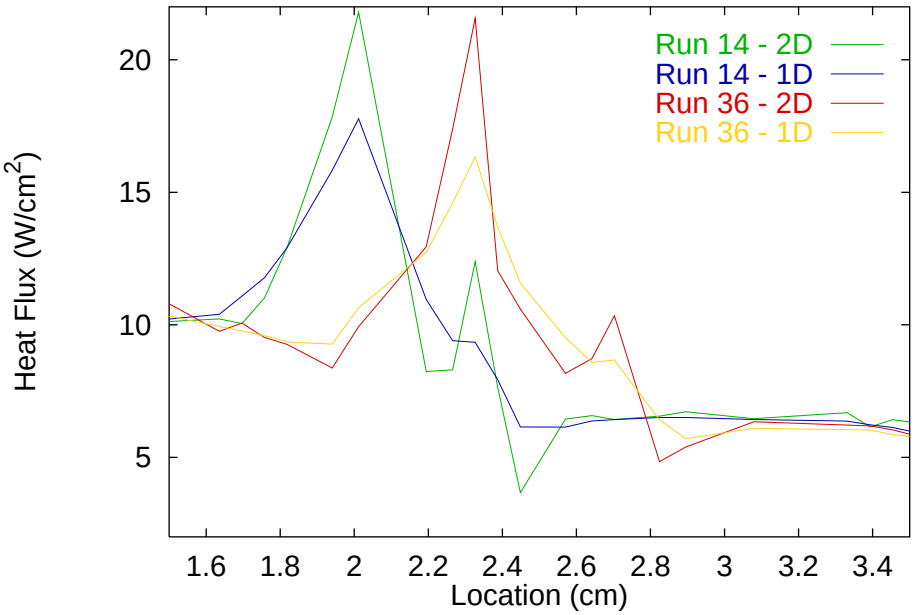

Fig. 7.26 One-dimensionaland two-dimensional estimates at peak heating location for Runs 14 and 36 (Macor model).

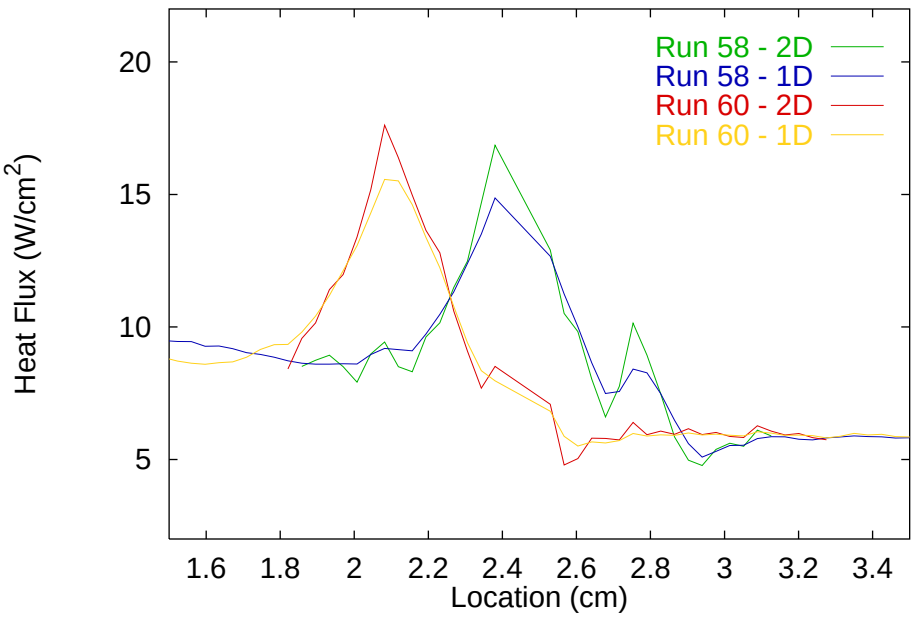

Fig. 7.27 One-dimensional and two-dimensional estimates at peak heating location for Runs 58 and 60 (Upilex model).

here in Table 7.4, are on the conservative side of the expected value. Realize, however, that the original estimate of peak heating assumes three-dimensional conduction which would indeed result in additional increase in peak heating levels over the two-dimensional analysis because the conduction is actually three-dimensional. It was also suggested by Berry and Nowak (1997) that the percent increase for the Upilex runs would be less than that of the Macor runs because Upilex is a better insulator. This feature was also confirmed. Therefore, the two-dimensional analysis performed here supports assumptions reported by Berry and Nowak (1996).

It should be noted at this point that the properties and geometry of the Upilex model are not known with a great deal of confidence. For example, the thickness and properties of the glue layer have not been established. Furthermore, the thermal properties of the Upilex layer is not known well. Therefore, for this analysis, the assumptions made by NASA LaRC (Berry and Nowak, 1996) were used here to provide a reasonable comparison. However, the values for the Upilex model are not expected to agree with the values of the Macor model. For details of the model, see Appendix C.

Table 7.4 The increase in peak heating estimate between one and two-dimensional approaches as a result of accounting for lateral conduction effects.

Model	Run	Increase in Peak
Macor	14	25%
	36	34%
Upilex	58	13%
	60	14%

Chapter 8

Summary and Conclusions

8.1 Description of Scope of Work

The goal of this work was to develop a methodology to accurately estimate heating rates from surface temperature measurements with reliability and stability. The estimation procedure should be able to handle unsteady and nonuniform characteristics in the data. Typical unsteadiness include noisy data, high frequency oscillations, step inputs and variations in measurement errors. Nonuniformity in heating rate is important in the presence of localized heating rates. Therefore, the estimation procedure should be able to handle lateral conduction effects as well.

8.2 Summary of Results

The one-dimensional analysis was designed to establish confidence in the proposed inverse technology's ability to resolve unsteady heating rates by comparing estimates from many traditional methods. It was found that Class 1 methods (those that are characterized by analytic solutions) tend to amplify measurement errors and other uncertainties in the experimental parameters. Even though Class 2 methods (characterized by numerical solutions) exhibit the same sensitivity problems as the Class 1 methods, perhaps their most significant drawback is the inability to resolve surface gradients accurately. The inverse methods (Class 3), on the other hand, were able to accurately capture transient phenomena such as sudden increases in measured temperatures and errant temperature readings. Furthermore, the family of inverse solutions were able to handle a wide range of situations such as dynamic

fluctuations and noisy data.

Within the family of Class 3 techniques, it was found that relatively little added bias was required to stabilize the solutions. In fact, traditional formulations of inverse problems were found to perform slightly below the proposed methodology (*Explicit-Regularization*) and the *Exact-Matching* technique. It is suggested that the *Explicit-Regularization* is the method of choice for the dynamic data for a wide range of situations. This method reported better performance measurement values (Table 6.1) of the methods examined with greater reliability and accuracy.

The two-dimensional analysis was primarily designed to demonstrate that lateral conduction can be significant in the presence of large localized heating rates, and to demonstrate that inverse technology can enhance data reduction models (and thereby the accuracy of the estimates) by addressing multi-dimensional conduction. The reported increases in peak heating rate in Table 7.4 establishes the validity of our treatment of lateral conduction effects.

8.3 Conclusions

From this analysis, the value of proper treatment of experimental data becomes apparent. We can conclude the following from the results:

- Measurement noise and other uncertainties can significantly affect data reduction solutions;
- Appropriate treatment of lateral conduction is crucial for situations of localized heating rates;
- Inverse technology provides a vehicle for reliable data reduction of large, unsteady and nonuniform heat fluxes from surface temperature measurements; and
- The proposed technique, the *Explicit-Regularization* method, performs better than the other methods for those situations presented here.

The inherent advantages of the inverse techniques (including the new *Explicit-Regularization* method) stems from the assumptions concerning the distribution of the boundary condition between discrete times or locations. Traditional methods typically assume some functional

relationship of the temperature between measurements (Section 3.1.2). The inverse techniques, on the other hand, make an assumption about the functional relationship of the heat flux between measurement times and locations (Chapter 4). Because the sensitivity of the heat flux to uncertainties in the measurements is large, any additional uncertainties in the assumption about the temperature between measurements will become amplified in the solution. Inversely, the sensitivity of the boundary temperature to the flux estimate is small. Therefore, any uncertainty in the flux distribution between measurement times and locations does not significantly affect the solution. For this reason, extra care must be taken to resolve numerical temperatures extremely accurately for Class 2 techniques. Whereas Class 3 techniques are not as affected by slight deviations in the conduction solution.

In a practical sense, these improvements in accuracy and confidence must be weighed in against their cost. Given in Table 8.1 is a rough estimate of the relative costs for the traditional techniques and the new approach. These reported times are generalizations of the amount of cpu time consumed by all the methods in a particular class. The computer was a 200 MHz Pentium Pro except for the two-dimensional analysis which was performed on a remotely located SGI. (The two-dimensional values have been adjusted for the difference in cpu.) Note that Class 3 methods (including the *Explicit-Regularization* method) are the only approaches capable of performing a true two-dimensional analysis. The benchmark test case consisted of 90 sensors and 200 time steps.

The advancement of this data reduction technology relates to several advantages to the shock interaction community as well as to those who reduce surface temperature measurements. The development of the *Explicit-Regularization* method allows estimation of heating rates with increased confidence without an increase in cost over Class 2 methods. All the issues that affect accuracy and stability can adequately be addressed by the new

Table 8.1 Approximate relative costs of the different methods to perform the indicated analysis.

Method	1 sensor one-dimensional	90 sensors one-dimensional	90 sensors two-dimensional
Class 1	1 sec	30 sec	—
Class 2	1 min	90 min	—
Class 3	2 min	180 min	30 hours
Explicit-Regularization	1 min	90 min	20 hours

technique.

Perhaps more importantly, however, the scope of possible experiments has been broadened. For example, it was shown how critical the sensor spacing can be for long duration experiments. As the sensor density increases, the data reduction problem becomes more difficult because of lateral conduction effects. With the proposed multi-dimensional *Explicit-Regularization* approach, lateral conduction can be successfully accounted for. This advancement opens the door for longer duration experiments with even higher data densities and complex geometries such as thermographic phosphor measurements on complex models.

The inverse methods, specifically the *Explicit-Regularization* method, can provide enhanced reliability and accuracy of data reduction problems involving dynamic surface temperature measurements.

Bibliography

Bass, B. R., (1980), "Application of the Finite Element Method to the Nonlinear Inverse Heat Conduction Problem Using Beck's Second Method," *ASME Journal of Engineering for Industry*, Vol. 102, pp. 168–176.

Baumeister, J. and Reinhardt, H. J., (1987), "On the Approximate Solution of a Two-Dimensional Inverse Heat Conduction Problem," in Proceedings of the *Inverse and Ill-Posed Problems*, eds. H. G. Engl and C. W. Groetsch, Academic Press, Orlando, Florida, pp. 325–344.

Beck, J. V., (1968), "Surface Heat Flux Determination Using an Integral Method," *Nuclear Engineering and Design*, Vol. 7, pp. 170–178.

Beck, J. V., (1970), "Non-linear Estimation Applied to the Non-Linear IHCP," *Journal of Heat Transfer*, Vol. 13, pp. 703–716.

Beck, J. V., (1979), "Criteria for Comparison of Methods of Solution of the Inverse Heat Condution Problem," *Nuclear Engineering and Design*, Vol. 53, pp. 11–22.

Beck, J. V., Blackwell, B., and St. Clair Jr., C. R., (1985), *Inverse Heat Conduction: Ill-Posed Problems*, John Wiley and Sons, Inc., New York, N.Y.

Beck, J. V. and Litkouhi, B., (1982), "Efficient Sequential Solution of the Non-linear IHCP," *Journal of Numerical Heat Transfer*, Vol. 5, pp. 275–286.

Berry, S. A. and Nowak, R. J., (1996), "Effects of Fin Leading Edge Sweep on Shock-Shock Interaction at Mach 6," *AIAA Conference Paper 96-0230.*

Berry, S. A. and Nowak, R. J., (1997), "Effects of Fin Leading Edge Sweep on Shock-

Shock Interaction at Mach 6," *AIAA Journal of Spacecraft and Rockets*, Vol. 34, No. 4, pp. 416–425.

Burcham, Jr., F. W. and Nugent, J., (1970), "Local Flow Field Around a Pylon-Mounted Dummy Ramjet Engine on the X-15-2 Airplane for Mach Numbers from 2.0 to 6.7," Tech. Report, NASA TN D-5638, N70-18035.

Buttsworth, D. R. and Jones, T. V., (1997a), "A Fast-Response High Spatial Resolution Total Temperature Probe Using a Pulsed Heating Technique," in Proceedings of the *International Gas Turbine & Aeroengine Congress and Exposition*, ASME, Orlando, Florida.

Buttsworth, D. R. and Jones, T. V., (1997b), "Radial Conduction Effects in Transient Heat Transfer Experiments," *The Aeronautical Journal*, Vol. 101, No. 1005, pp. 209–212.

Carslaw, H. S. and Jaeger, J. C., (1959), *Conduction of Heat in Solids*, Oxford University Press, Inc., New York.

Chadwick, K. M., (1997), "Stagnation Heat Transfer Measurement Techniques in Hypersonic Shock Tunnel Flows over Spherical Segments," in Proceedings of the *32nd AIAA Thermophysics Conference*, No. 97-2439, Atlanta, Georgia.

Cook, W. J., (1970a), "Determination of Heat Transfer Rates from Transient Surface Temperature Measurements," *AIAA Journal*, Vol. 8, No. 7, pp. 1366–1368.

Cook, W. J., (1970b), "Unsteady Heat Transfer to a Semi-Infinite Solid with Arbitrary Surface Temperature History and Variable Thermal Properties," Tech. Report, Engineering Research Institute TR ISU-ERI-Ames 67500, Iowa State University, Ames, Iowa.

Cook, W. J. and Felderman, E. J., (1966), "Reduction of data from Thin Film Heat Transfer Gages: A Concise Numerical Technique," *AIAA Journal*, Vol. 4, No. 3, pp. 561–562.

Diller, T. E., (1996), "Methods of Determining Heat Flux from Temperature Measurements," , pp. 357–369.

Diller, T. E. and Kidd, C. T., (1997), "Evaluation of Numerical Methods for Determining Heat Flux with a Null Point Calorimeter," in Proceedings of the *42nd International Instrumentation Symposium*, ISA, Research Triangle Park, NC, pp. 251–262.

Dunn, M. G., George, W. K., Rae, W. J., Woodward, S. H., Moller, J. C., and Seymour, P. J., (1986), "Heat Flux Measurements for the Rotor of a Full-Stage Turbine: Part II: Description of Analysis Technique and Typical Time-Resolved Measurements," *ASME Journal of Turbomachinery*, Vol. 108, No. 1, pp. 98–107.

Edney, B. E., (1968a), "Anomalous Heat Transfer and Pressure Distributions on Blunt Bodies at Hypersonic Speeds in the Presence of an Impinging Shock," Tech. Report, The Aeronautical Research Institute of Sweden, FFA Report 115, N68-33668.

Edney, B. E., (1968b), "Effects of shock impingement on the heat transfer around blunt bodies," *AIAA Journal*, Vol. 6, pp. 15–21, A68-14869.

Edney, B. E., (1970), "Shock Interference Heating and the Space Shuttle," Tech. Report, NASA TM X-52876, N70-37840, see also N70-37826.

Ehrich, F. F., (1954), "Differentiation of Experimental Data Using Least Squares Fitting," *Journal of the Aeronautical Sciences*, Vol. 22, pp. 133–134.

George, W. K., Rae, W. J., Seymour, P. J., and Sonnenmeier, J. R., (1987), "An Evaluation of Analog and Numerical Techniques for Unsteady Heat Transfer Measurements with Thin Film Guages in Transient Facilities," in Proceedings of the *ASME/JSME Thermal Engineering Joint Conference*, Vol. 2, Honolulu, HI, pp. 611–617.

Gladden, H. J. and Melis, M. E., (1993), "Hypersonic Engine Component Experiments in a High Heat Flux, Supersonic Flow Environment," in Proceedings of the *International Symposium on Optical Applied Science and Engineering*, Society of Photo-Optical Instrumentation Engineers, San Diego, California, pp. 472–485.

Gladden, H. J. and Melis, M. E., (1994), "Hypersonic Leading Edge Experiments in a High Heat Flux, Supersonic Flow Environment," in Proceedings of the *ASME Winter Annual Meeting*, Chicago, Illinois, N95-14299.

Glass, C. E., (1989), "Experimental Study of Pressure and Heating Rate on a Swept Cylindrical Leading Edge Resulting from Swept Shock Wave Interference," M.S. Thesis, Old Dominion University, Norfolk, Virginia.

Glass, C. E., Wieting, A. R., and Holden, M. S., (1989), "Effect of Leading Edge Sweep on Shock-Shock Interference at Mach 8," in Proceedings of the *27th AIAA Aerospace Sciences Meeting*, No. 89-0271, Reno, Nevada, 89A25229.

Guo, L. and Murio, D. A., (1991), "A Mollified Space-Marching Finite Difference Algorithm for the Two-Dimensional Inverse Heat Conduction Problem with Slab Symmetry," *Inverse Problems*, Vol. 7, pp. 247–259.

Hains, F. D. and Keyes, J. W., (1972), "Shock Interference Heating in Hypersonic Flows," *AIAA Conference Paper 72-78*, A72-16805.

Hedlund, E. R., Hill, F. A. F., Ragsdale, W. C., and Voisinet, R. L. P., (1980), "Heat Transfer Testing in the NWSC Hypervelocity Wind Tunnel Utilizing Co-Axial Surface Thermocouples," Tech. Report, Naval Surface Weapons Center MP 80-15I.

Henshall, B. D. and Schultz, D. L., (1959), "Some Notes on the Use of Resistance Thermometers," *Aeronautics Research Council*, Vol. 408.

Hiers, R. S. and Loubsky, W. J., (1967), "Effects of Shock-Wave Impingement on the Heat Transfer on a Cylindrical Leading Edge," Tech. Report, NASA TN D-3859, N67-17798.

Holden, M. S., (1975), "Experimental facilities and measurement techniques," *Advisory Group for Aerospace Research and Development*, pp. 44–48, N75-32005.

Holden, M. S. and Kolly, J. M., (1995), "Measurements of Heating in Regions of Shock/Shock Interaction in Hypersonic Flow," *AIAA Conference Paper 95-0640*.

Holden, M. S., Moselle, J. R., and Lee, J., (1991), "Studies of Aerothermal Loads Generated in Regions of Shock/Shock Interaction in Hypersonic Flow," Tech. Report, NASA Contractor Report 181893, N92-11319.

Holden, M. S., Wieting, A. R., Moselle, J. R., and Glass, C., (1988), "Studies of Aerothermal Loads Generated in Regions of Shock/Shock Interaction in Hypersonic Flow," *AIAA Conference Paper 88-0477*, A88-22352.

Hollis, B. R., (1995), "User's Manual for the One-Dimensional Hypersonic Experimental Aero-Thermodynamic (1DHEAT) Data Reduction Code," Tech. Report, NASA Contractor Report 4691.

Imber, M., (1974), "Temperature Extrapolation Mechanism for Two-Dimensional Heat Flow," *AIAA Journal*, Vol. 7, pp. 1089–1093.

Jones, J. J., (1959), "Shock Tube Heat Transfer Measurments on Inner Surface of a Cylinder (Simulating a Flat Plate) for Stagnation Temperature Range 4100 °R to 8300 °R," Tech. Report, NASA TN-D-54.

Kendall, D. N. and and, W. P. D., (1966), "Heat Transfer Measurements in a Hot Shot Wind Tunnel," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-3, No. 4, pp. 596–603.

Keyes, J. W. and Hains, F. D., (1973), "Analytical and experimental studies of shock interference heating in hypersonic flows," Tech. Report, NASA TN D-7139, N73-25283.

Keyes, J. W. and Morris, D. J., (1972), "Correlations of peak heating in shock interference regions at hypersonic speeds," *Journal of Spacecraft and Rockets*, Vol. 9, pp. 197, A72-41309.

Melis, M. E., Gladden, H. J., Schubert, J. F., Westfall, L. J., and Trimarchi, P. A., (1989), "A Unique Interdisciplinary Research Effort To Support Cowl Lip Technology Development for Hypersonic Applications," Tech. Report, NASA Technical Paper 2876.

Micol, J. R., (1995), "Hypersonic Aerodynamic/Aerothermodynamic Testing Capabilities at Langley Research Center: Aerothermodynamic Facilities Complex," in Proceedings of the *30th AIAA Thermophysics Conference*, No. 95-2107.

Miller, C. G., (1981), "Comparison of Thin-Film Resistance Heat Transfer Gages with Thin-Skin Transient Calorimeter Gages in Conventional Hypersonic Wind Tunnels," Tech. Report, NASA TM-83197.

Miller, C. G., (1990), "Langley Hypersonic Aerodynamic/Aerothermodynamic Testing Capabilities – Present and Future," *AIAA Conference Paper 90-1376.*

Morris, D. J. and Keyes, J. W., (1973), "Computer programs for predicting supersonic and hypersonic interference flow fields and heating," Tech. Report, NASA TM X-2725, N73-25285.

Murio, D. A., (1985), "On the Characterization of the Solution of the Inverse Heat Conduction Problem," in Proceedings of the *ASME Winter Annual Meeting*, Miami Beach, Florida, USA 85-WA/HT-41, 7p.

Neumann, R. D., (1996), "High Speed Shock Impingement Studies Using Innovative Movement of Model Components to Expedite Testing in Traditional Wind Tunnel Facilities," *AIAA Conference Paper 96-2192.*

Patankar, S. V., (1980), *Numerical Heat Transfer and Fluid Flow*, Taylor & Francis.

Prabhu, R. K., (1994), "An Implementation of a Chemical and Thermal Nonequilibrium Flow Solver on Unstructured Meshes and Application to Blunt Bodies," Tech. Report, NASA Contractor Report 194967, N95-12619.

Schultz, D. L. and Jones, T. V., (1973), "Heat Transfer Measurements in Short Duration Hypersonic Facilities," Tech. Report 165, Advisory Group for Aerospace Research and Development No. 165.

Scott, E. P. and Beck, J. V., (1987), "Analysis of Order of the Sequential Regularization Solutions of Inverse Heat Conduction Problems," *ASME Journal of Heat Transfer*, Vol. 111, pp. 218–224.

Sparrow, E. M., Haji-Sheikh, A., and Lundgren, T. S., (1964), "The Inverse Problem in Transient Heat Conduction," *ASME Journal of Applied Mechanics*, Vol. 86, pp. 369–375.

Stolz, Jr., G., (1960), "Numerical Solutions to an Inverse Problem of Heat Conduction for Simple Shapes," *ASME Journal of Heat Transfer*, Vol. 82, pp. 20–26.

Tannehill, J. C., Holst, T. L., Rakich, J. V., and Keyes, J. W., (1976), "Comparison of a two-dimensional shock impingement computation with experiment," *AIAA Journal of Thermophysics*, Vol. 14, pp. 539–541, A76-33724.

Tikhonov, A. N. and Arsenin, V. Y., (1977), *Solutions of Ill-Posed Problems*, V. H. Winston & Sons, Washington, D. C.

Trucco, R. E. and Tamango, J., (1990), "Elimination of an External Noise Spike on the Thin Film Temperature Traces," Tech. Report, GASL Technical Memorandum 240.

Vidal, R. J., (1956), "Model Instrumentation Techniques for Heat Transfer and Force Measurements in a Hypersonic Shock Tunnel," Tech. Report, Cornell Aeronautic Laboratory AD-917-A-1.

Walker, D. G. and Scott, E. P., (1995), "One-Dimensional Heat Flux History Estimation from Discrete Temperature Measurements," in Proceedings of the *ASME International Symposium and Exposition*, San Francisco, California.

Walker, D. G. and Scott, E. P., (1997), "A Method for Improving Two Dimensional High Heat Flux Estimates from Surface Temperature Measurments," in Proceedings of the *32nd AIAA Thermophysics Conference*, No. 97-2574.

Watts, J. D., (1968a), "Flight Experience with Shock Impingement and Interface Heating on the X–15–2 Research Airplane," Tech. Report, NASA TM X-1669, N92-70863.

Watts, J. D., (1968b), "Heat Transfer Effects of Surface protuberances on X–15 airplane," Tech. Report, NASA TM X-1566, N75-70033.

Whetstone, D. W., (1983), *Engineering Analysis Language*, Engineering Information Systems, Inc., San Jose, CA.

Wieting, A. R. and Holden, M. S., (1987), "Experimental Study of Shock Wave Interference Heating on a Cylindrical Leading Edge at Mach 6 and 8," *AIAA Conference Paper 87-1511.*

Wieting, A. R., Thareja, R. R., Stewart, J. R., and Morgan, K., (1988), "Viscous Hypersonic Unstructured Grid Computation of Shock–on–Shock Interaction on Blunt Cowl Lips," in Proceedings of the *Fourth National Aerospace Plane Technology Symposium.*

Appendix A

Alternate Sensitivity Calculation Method

When using a finite element technique, the calculation of the sensitivity matrix $\mathbf{X}$ can be made more efficient by examining the matrix equations for a typical finite element solution methodology. Note that other numerical approaches could be manipulated in a similar fashion to yield the sensitivity; finite element formulations are used here because of the convenience of the method for the forward conduction solution and the generality of the derivation.

It was found that the calculation of the sensitivity coefficients was extremely costly when found using differences (Beck, 1970) as was done for the previous analysis. By employing a linearizer to the non-linear problem (Beck and Litkouhi, 1982), the solution can be sped up considerably.

For our finite element solution method, the difference equations are given as

$$\mathbf{KT} + \mathbf{C}\dot{\mathbf{T}} = \mathbf{F}\,, \tag{A.1}$$

where the properties are embedded in the assembled stiffness (K) and capacitance matrices (C). The boundary and source loads appear in the matrix F. The sensitivity matrix is defined as the derivative of the temperature with respect to the boundary flux and is given as

$$\mathbf{X} = \frac{\partial \mathbf{T}}{\partial \mathbf{q}}\,. \tag{A.2}$$

Now we make the simplifying assumption suggested by Beck and Litkouhi (1982), that the coefficient matrices do not change with temperature for small changes in temperature. This will be true for small time steps and for situations when the properties are only a weak function of temperature. Now simply differentiate the difference equation (Eq. (A.1)) with respect to the boundaryt flux realizing that the load vector will reduce to unity when the only load is the boundary condition. The resulting set of equations, given by

$$\mathbf{KX} + \mathbf{C}\dot{\mathbf{X}} = \mathbf{1}\,, \tag{A.3}$$

appears to be remarkably similar to the original. Also realize that the initial distribution for this solution must be zero since $\partial \mathbf{T_o}/\partial \mathbf{q} = \mathbf{0}$. The new difference equation can be solved relativelly cheaply compared to the conduction solution because the stiffness matrix and the capacitance matrix have already been calculated for the forward conduction solution. In this manner the sensitivities can be found through one additional (simplified) forward solution rather than many full blown conduction solutions that were needed for the solution method used for the analysis in this work.

Appendix B

Conduction Models

B.1 Thermal Properties

The temperature dependent thermal properties that were used throughout the analysis are given by Hollis (1995) and are repeated here for completeness. The substrate for the unsteady analysis consists of Pyrex exclusively, whereas, Macor and Upilex are used for the substrate in the nonuniform analysis. The formulations for the pertinent properties are given in Appendix F for english and si units. Note that the formulas have been adapted for computational speed. In several instances, the property value at a particular temperature is needed such as for reference temperatures and for constant property values. For convenience, these are tabulated here.

Material	Temperature	Property	Value[a]
Pyrex	524 R	ρ	$4.321\,\mathrm{slug/ft^3}$
		c_p	$5.776\,\mathrm{Btu/slug\,^\circ R}$
		k	$0.002143\,\mathrm{Btu/ft\,sec\,R}$
		β'	$0.001294\,\mathrm{1/R}$
Macor	300 K	ρ	$2544\,\mathrm{kg/m^3}$
		c_p	$733.7\,\mathrm{J/kg\,K}$
		k	$2.454\,\mathrm{W/m\,K}$

[a]The set of units corresponds to the units of the original data

B.2 One-Dimensional Model

The unsteady analysis was performed using the Calspan data and corresponding one-dimensional test cases. The Calspan model consisted of a cylindrical disk (3 inch or 7.62 cm diameter) manufactured from Pyrex. For the one-dimensional analysis, however, the cylinder was treated as a semi-infinite slab. This assumption proved to be valid since the penetration depth is small compared to the diameter of the cylinder. If t_f is the final time, then the penetration at that time can be given conservatively as

$$\eta_f = \sqrt{\alpha_o t_f} \approx 0.011\,\text{inch}\,, \tag{B.1}$$

if the thermal diffusivity $\alpha_o = k/\rho\, c_p$ is found using the reference properties for Pyrex given previously.

B.3 Two-Dimensional Models

The two-dimensional model also consists of a cylinder but with two different geometries. For the one-dimensional estimation procedure, the model is treated as a semi-infinite slab which may not be an adequate model for this case. Realize that the experimental time is a good bit longer than in the unsteady analysis, and the diameter of the cylinder is smaller (1.27 cm diameter). The penetration depth for this case becomes

$$\eta_f = \sqrt{\alpha_o t_f} \approx 0.3\,\text{cm}\,. \tag{B.2}$$

It was suggested by Buttsworth and Jones (1997b) that the curvature becomes significant in estimating fluxes, so for the two-dimensional methods, the cylinder was modeled with curvilinear coordinates with the specified radius even though there were no nodes in the angular direction. The two-dimensional mesh then was oriented into the material toward the centerline and along the model.

The Upilex model was similar to the Macor model except now there are additional conduction layers on the surface. A 2 mil (0.0051 cm) layer of Upilex and a 2 mil layer of high temperature epoxy were modelled on the surface of the Macor. The thermal properties of the glue were unknown, so to duplicate the analysis performed at NASA LaRC, the properties were assumed to be the same as the Upilex layer.

Appendix C

Verification of Conduction Solutions

There are four numerical conduction solutions methods given in the following list, which need to be verified.

1. one-dimensional, finite difference, heat flux boundary (FD)
2. one-dimensional, finite difference, temperature boundary
3. one-dimensional, finite element, heat flux boundary (FE)
4. two-dimensional, finite element, heat flux boundary

Note that all methods employ a semi-infinite slab assumption.

The canonical verification of conduction solutions is a comparison to an analytic solution such as a step flux. An analytic solution to this case was found for this comparison. However, verification of the solution using dynamic data was also needed. An analytic solution did not exist so we were required to compare two numerical solutions. Finally, to establish convergence given temperature dependent properties, two numerical solutions were compared, once again.

C.1 Standard Evaluation

The first three methods were used to solve for conduction of the unsteady cases which were designed to model the Calspan data (see Section 2.3 for a description of the experiment

and Section 5.2 for a description of the types of data). To test the validity of these solution methods, they were first compared to an analytic solution of a step flux (as given in Section 5.2.2). First we examine the interior temperature distribution at the time the flux is turned off (0.007 sec) and the time at which the experiment ends (0.01 sec). All temperatures here should be less the 0.1K from the analytic solution and the temperature at the interior surface should be the initial temperature AND should be insulated. The next check involves examining the surface temperature history. Again, all temperatures should not differ from the analytic solution by more than 0.1K.

Since finite difference methods are fully implicit, we expect a stable solution but not necessarily a physically realistic solution. We must fix the integration time step in accordance with the explicit stability criterion

$$\frac{\alpha \Delta t}{\Delta x^2} < \frac{1}{2} \tag{C.1}$$

where Δx is the nodal spacing and Δt is the integration time step, to obtain an accurate solution. In this formulation, α represents the thermal diffusivity (a material property) evaluated at the initial temperature and the nodal spacing can be found from the penetration depth η and the number of nodes N ($\Delta x = \eta/N$).

The numerical and analytical temperature distribution show that the penetration depth for the experiment is less than the selected domain of 0.06 cm. Also, the number of nodes that result in a converged solution for both distributions is forty. However, fifty nodes were used to ensure a converged solution.

The surface temperature history of the analytic solution along with the numerical solution shows that again, forty nodes is adequate to resolve the surface temperatures within 1K of the analytic result. The fifty node case is also shown since this was the number of nodes used. Since the solution is most critical at the location of the step and since this is the region that will result in the most significant errors, the region was scrutinized for accuracy. Using the same time scales, Fig. C.3 examines the step, and finds that the temperatures between the analytic and numerical solutions agree well.

C.2 Dynamic Data

The same analysis was applied to the high frequency oscillating flux as described in Section 5.2.5. It was found that this particular data require slightly more nodes to reach the

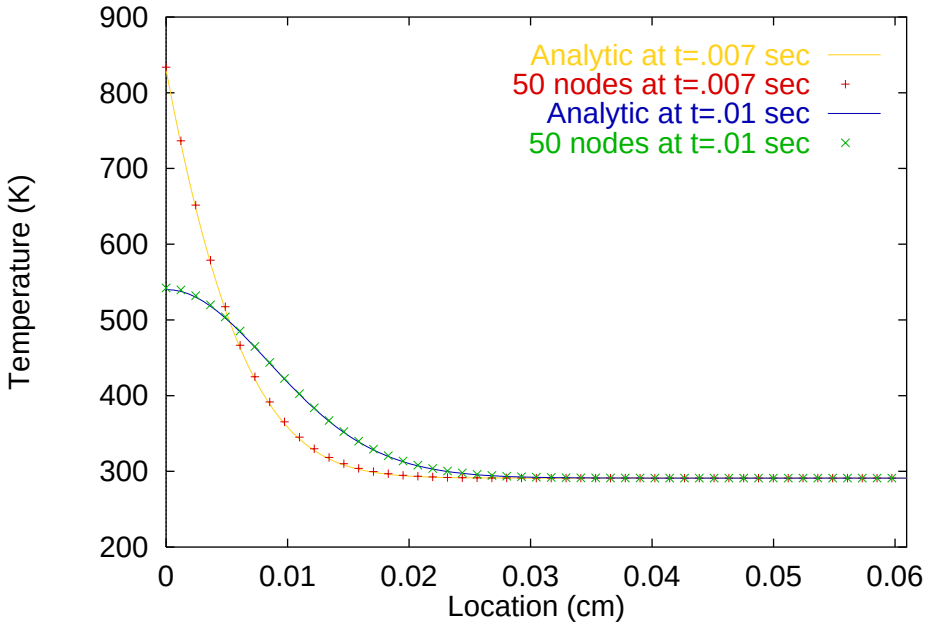

Fig. C.1 Analytical and numerical distribution for step flux

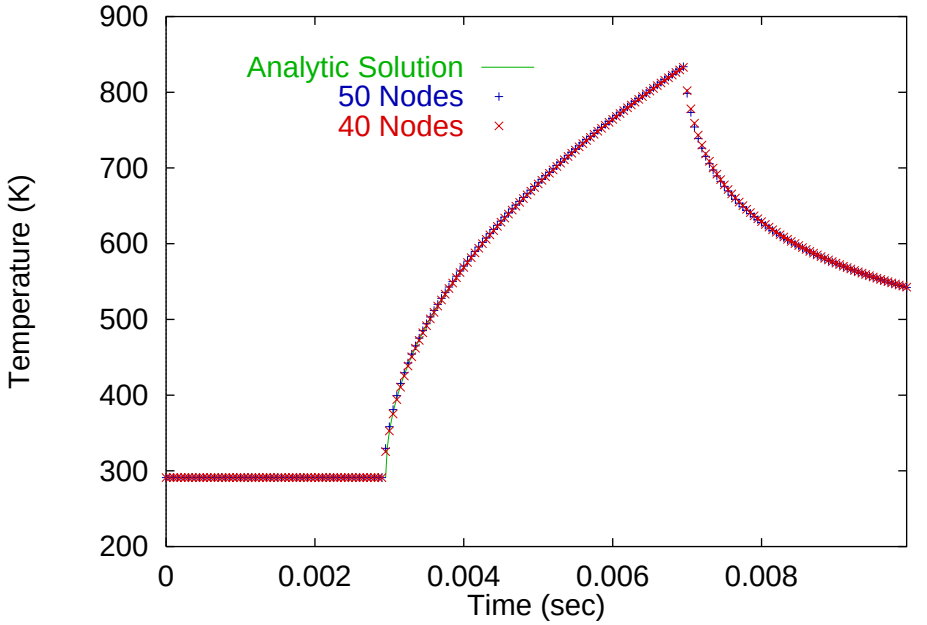

Fig. C.2 Analytical and numerical surface temperature history

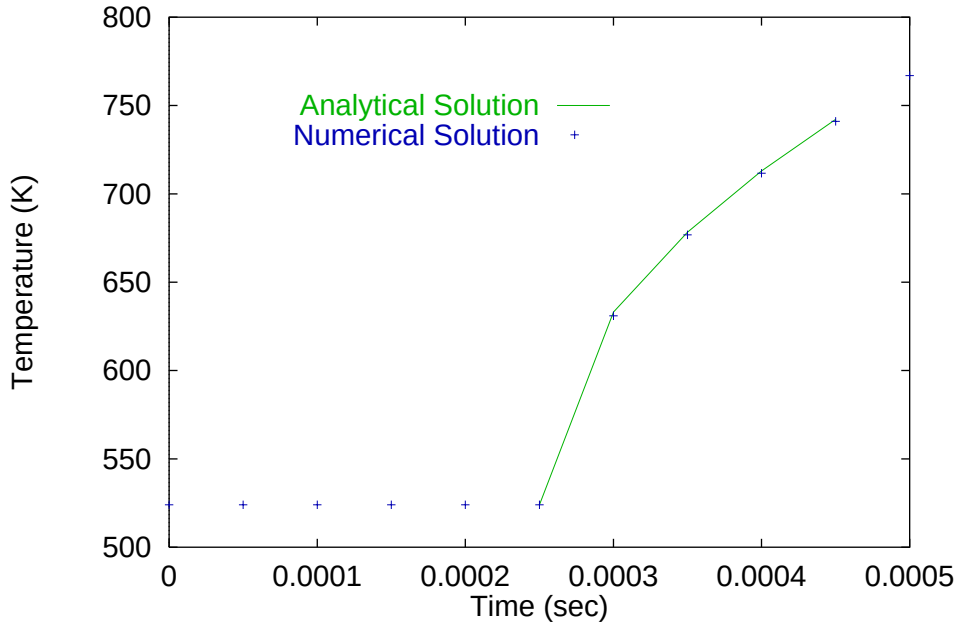

Fig. C.3 Enlargement of analytical and numerical surface temperature history.

converged solution. However, it was found that the solution became independent of the grid spacing at 50 nodes.

Since an analytic solution was not available, we compared the FD solution to a FE solution. The FE code allows complete flexibility in regards to grid generation, integration time step, and explicit/implicit formulation. As suggested by Whetstone (1983), a Galerkin approach ($\beta = 2/3$) was used, which is between a Crank-Nicolson and a fully implicit scheme. The grid was designed to be identical to the FD grid which proved to be successful. The integration time step, however, was taken to be a value suggested by a processor in the FE code. This value turned out to be somewhat smaller than the time step used in the FD method. Upon comparison of the two methods, it was found that the solution of the FD method became independent of the integration time step as it approached the value suggested by the FE method.

The step flux provided a baseline to begin the verification. The high frequency case, however, required us to refine the methods to accommodate dynamic situations. Table C.1 itemizes the parameter necessary to ensure a converged solution in both cases. It should be noted that all values are somewhat conservative and that the requirements of the FE technique could have been relaxed compared to the FD technique.

Table C.1 Parameters used in the FD and FE techniques to ensure a converged conduction solution.

Parameter	Value
Penetration depth	0.06 cm
Number of nodes	50
Integration time step	$0.000150/\alpha$

C.3 Variable Temperature Properties

The Calspan test cases were used Pyrex as the conduction medium. The properties do not vary significantly over small ranges (Hollis, 1995), so it was assumed that the same convergence parameters could be used. Again there is no analytic solution, so upon comparison of the two numerical solutions, this assumption was found to be valid.

C.4 Multi-Dimensional Grid

The two-dimensional estimation procedure uses a two-dimensional conduction mesh for the two-dimensional FE solution. To verify this case, two approaches were used. A two dimensional analytic solution was devised with a step in space and constant flux in time to find the appropriate lateral grid spacing. Similar to the one-dimensional convergence evaluation, a one-dimensional flux was applied to verify that the resulting two-dimensional distribution was in fact one-dimensional.

Since this two-dimensional FE approach was only used for the conduction solution of the NASA data, the material properties, time scales, and geometries were significantly different. All convergent parameters had to be re-evaluated. For the Macor model, constant properties were assumed to compare the single conduction layer with an analytic solution. The grid was arranged geometrically so that the total number of computational nodes could be reduced. The grid spacing was smaller near the surface and progressively became larger towards the interior surface as given by the nodal locations as

$$x_i = x_s - \eta \cos\left[\frac{(i-1)}{N}\frac{\pi}{2}\right] . \tag{C.2}$$

Again, η is the penetration depth, N is the number of nodes and x_s is the distance to the surface (radius of the Macor cylinder). Because the large gradients only exist at the surface, this scheme demonstrated how computational time could be shortened without loss

of accuracy. The lateral node spacing was taken to be roughly the same as the distance between the first two nodes into the material (calculation schemes in the FE did not allow exact agreement). Table C.2 shows the results of the two-dimensional convergence study. These parameters allowed the method to capture any transient or spatial phenomenon accurately.

Table C.2 Parameters used in the two-dimensional FE technique to ensure a converged conduction solution.

Parameter	Value
Penetration depth	0.6 cm
Number of nodes	30
Lateral node spacing	0.001 cm
Integration time step	$0.001 sec$

Appendix D

Temperature Dependent Property Correction

To determine a temperature correction factor for a particular material, a numerical conduction solution using temperature dependent properties is compared to an analytic solution using properties evaluated at a reference temperature. (See Hollis (1995) for a description of this procedure.) For situations were the variation in the property is linear and slight with temperature, the approximation can be adequate. However, for real materials such as Pyrex, Upilex and Macor that are used extensively in this analysis, the properties generally must be described using higher ordered polynomials. In these cases, a universally applicable correction factor can not be found.

The reason for the lack of generality of the correction is due to the fact that the correction is a function not only of temperature on the surface, but also of the current temperature distribution. In other words, the amount of conduction and storage within the material is affected by the temperature everywhere in the material. Therefore, the material's temperature history affects the amount of correction needed. For methods that employ this correction, it is assumed that the correction is a function of the surface temperature only. Therefore, we may see deviations from the expected amount of correction between a step in flux and an oscillating flux even when the surface temperatures are the same.

Appendix E

Two-Dimensional Inverse Code

These scripts, written in EAL (Engineering Analysis Language) contain routines to execute a two-dimensional *Function-Specification* method or a *Regularization* method. Note that the *Explicit-Regularization* method and the *Exact-Matching* method can be achieved by setting the number of simultaneous times to 1. The program was written in segments that were assembled at run time depending on the methodology used.

There are four types of modules needed to assemble a working program.

1. **Comment Header** This module is common to all programs. It merely gives pertinent information as to the required definitions, defined tables and libraries used.

2. **Model Description** This module defines the parameters of the model and generates the mesh. Also thermal property tables and boundary conditions are read into the tables at this point.

3. **Patch Definition** These modules define the type of patch used for sensitivity and flux update operations.

4. **Method Designation** These modules define the method to use to perform the optimization.

One module from each group must be selected and concatenated to generate a complete script which EAL will execute. The exception to this rule is the direct solver of course. In this case, the third module is unnecessary.

Section	Filename	Description
E.1	`comm.seg`	header file for each assembled program – describes libraries, variables and data structures used
E.2	`R14.seg`	defines the finite element model for the Macor model
E.3	`R58.seg`	defines the finite element model for the Upilex model
E.7	`flux.cnst.seg`	defines the flux correction scheme and corresponding sensitivity calculation for a 3 node patch
E.8	`flux.cnst5.seg`	defines the flux correction scheme and corresponding sensitivity calculation for a 5 node triangular patch
E.9	`flux.lin5.seg`	defines the flux correction scheme and corresponding sensitivity calculation for a 5 node constant patch
E.10	`flux.reg5.seg`	defines the flux correction scheme and corresponding sensitivity calculation for a 5 node patch (both triangular and constant patch logic are contained in this file)
E.4	`calc.dir.seg`	routine that performs the forward solution
E.6	`calc.reg5.seg`	routine that performs the *Regularization* method
E.5	`calc.inv5.seg`	routine that performs the *Function-Specification* method

E.1 Comment Module

```
$ EAL script automatically parsed from segments
$ COMMENT comm.seg
$ MODEL modl.seg
$ FLUX         flux.seg
$ CALC calc.seg
$
$ ================================================================
$
$ Segment COMMENT describes naming conventions used throughout all
$                 other segments
$
$ ================================================================
$
$ Variables: (1) - one layer model
$ LR - radius of model
$ LT - conduction depth (larger than penetration depth) (1)
$ LZ - length of model
$
$ Data Sets:
$ TRAN TEMP
$ TRAN TEMP 2
$ INIT TEMP
$ SURF TEMP
$ MEAS TEMP
$
$ Libraries:
$ 01 - General
$ 02 - ?
$
$ 05 - Binary property output
$ 06 - Binary node locations
$ 07 - Binary temperature measurements
$ 08 - Binary Flux estimate
$
$ 29 - Runtime datasets

$ Notes:
$ - All Segments must begin and end with a "$" (no white space)
$   or use sed '/^$/d' to remove blank lines and
$         dont worry bout it. This bizarre restriction is because
$         EAL is too braindead to allow blank lines for readability

!IL=11                  $Input Library
!OL=12                  $Binary Output Library
!TL=13                  $This will be the transient library
!HL=14                  $History Library (stored info)

!CY=-1.0                $Negative one for subtracting matrices

$ Here we have to ask EAL to get as much memory as it possibly can.
$ This amounts to 7M (Where the hell did that number come from ?!?)
$ Also note that this is hardly enough to reduce or prevent massive
$ disk I/O. (i.e. time consuming)

*XQT U1
!MAXCM=SSP(0,13) -1000000
*CMPARA(MAXCM="MAXCM")
*CM="MAXCM"

$ Here we turn off the annoying messages that EAL pukes up.

*XQT AUS
!MAXCM
*NOECHO 1,2,3,4
```

E.2 R14 Module

```
$ ================================================================
$
$ Segment GEOMETRY define geometry for 2-D shell
$                  (one conduction layer)
$
$ ================================================================


!LR=0.00635           $Radius of the model (m)
!LX=0.00635           $Depth of the layer (m) should be larger than the
$                     $penetration depth

$  Define number of elements and nodes in the z direction


!F=1.0              $Integration scheme


$ ================================================================
```

```
$
$ Segment MODEL define properties, input data, initial values
$ modl.r14.seg
$
$ ================================================================

$ ----------------------------------------------------------------
$
$  Section PROP - thermal property data
$
$ ----------------------------------------------------------------

*XQT AUS

$ Read in the varaible property data (T, rho, c, kxx)
$ NVE, the number of property entries must be defined in the data file

*TF OPEN 3'varbprop
*TF READ 3
*TF CLOSE 3
$
$ NI=9 indicates that nine variables can be entered to determine k
$ (T, rho, c, kxx, kyy, kzz, kxy, kzy, kzx).
$
$ For COND PROP n, n is the material number
$ For SOUR K21 n, n is the group number

$ The table of properties read in contain (T, rho, c, kxx). These
$ are being transferred to COND PROP

 DEFINE C =  2 VARB PROP
 TABLE(NI=9,NJ="NVE"): COND PROP 1
 TRANSFER(SOURCE=C,DSKIP=3,ILIM=6,JLIM="NVE",OPER=XSUM)

!DELT = 0.001
!DELK = 0.02

$ Thermal property table for the flux source.

 TABLE(NI=9,NJ=1): COND PROP 2: I = 3,4
 J=1: 0.0,0.0
*XQT DCU
 PRINT 1 COND PROP 1
 XCOPY 1, 5 COND PROP 1
```

```
$ ----------------------------------------------------------------
$
$  Section SLOC - from sensor location, get nodes for sensors
$
$ ----------------------------------------------------------------

*XQT U1

*(29 MK SLOC) SLOC

*XQT AUS

TABLE(NI=1,NJ="NS",TYPE=0): MINT NODE

!SL1 = DS 1, 1, 1, (SENS LOCS)
!SL2 = DS 1, "NS", 1 (SENS LOCS)

!LZ = SL2 - SL1
!LEZ = LZ / NEZ

!SCT = 0
*LABEL 621
    !SCT = SCT + 1

    !CSL = DS 1, "SCT", 1 (SENS LOCS)
    !SNN = IFIX( (CSL - SL1)/(SL2 - SL1)*FLOAT(NEZ) ) + 1

    TABLE,U(TYPE=0): MINT NODE: I=1: J="SCT": "SNN"

    !ENDS = SCT - NS
*JLZ(ENDS, 621)

*RETURN
*SLOC

$ ----------------------------------------------------------------
$
$  Section INPT - determine the measured temperature data
$                 initialize data sets
$
$ ----------------------------------------------------------------

*XQT AUS

!NR = 1       $number of time steps to be considered at once
```

```
$ Note that NTS is in sourtime and NS is in senslocs
$ they must be read before meastemp

*TF OPEN 4'sourtime.eal
*TF READ 4
*TF CLOSE 4

*TF OPEN 7'senslocs.eal
*TF READ 7
*TF CLOSE 7

*TF OPEN 8'meastemp.eal
*TF READ 8
*TF CLOSE 8

$ Based on the input read, we can define the rest of the parms

!SL0 = DS 1, 1, 1 (1 SENS LOCS)
!SLN = DS 1, "NS", 1 (1 SENS LOCS)

$ this value was developed through trial and error so that
$ the appropriate number of nodes exists per run
!NEZ = IFIX(SLN -SL0 *6000)

$!NEZ=(NS-1)*10     $Number of elements along the model (z)
!NNZ=NEZ+1         $Number of nodes

!NEX=30            $Number of elements into the cylinder (r)
!NNX=NEX+1         $Number of nodes

!NT=NNZ*NNX        $Total number of nodes in the complete model

!N1=1              $Beginning node for shell
!N2=1              $Beginning node for flux elements

!X1=0.0            $Radial Position of node #1

!LEX=LX/NEX        $Length of an element in x direction

!DX=LX/(NEX*(NEX+1))

!NTI=NS*NR

*DCALL (29 MK SLOC)
 TABLE(NI="NTI",NJ=1): Y
 TABLE(NI="NTI",NJ=1): T
 TABLE(NI="NTI",NJ=1): TDQ
$
!TEMI = DS 1, 1, 1 ("HL" MEAS TEMP)
 TABLE(NI=1,NJ="NT"): "HL" TRAN TEMP 2: J=1,"NT": "TEMI"
 TABLE(NI=1,NJ="NT"): "HL" INIT TEMP
 TABLE(NI="NNZ",NJ="NTS"): "HL" SURF TEMP
$
 DEFINE TT2 = "HL" TRAN TEMP 2
!S1 = N2 - 1
 TABLE,U(TYPE=-2): "HL" SURF TEMP
 TRANSFER(SOURCE=TT2,SBASE="S1",ILIM="NNZ",OPER=XSUM)
$
*XQT DCU
 PRINT 1 MINT NODE
 PRINT "HL" MEAS TEMP
 PRINT 1 SOUR TIME
 XCOPY "HL",7 MEAS TEMP
 XCOPY 1, 9 MINT NODE
$
$ ----------------------------------------------------------------
$
$  Section BOUN - initilaize boundary conditions
$
$ ----------------------------------------------------------------
$
*XQT AUS
$
 TABLE(NI="NEZ",NJ="NTS"): SOUR K21 1
$
$ The table of fluxes just created must be put into the form such that
$ J is the number of elements that have a source value
$ BLOCK is the number of time steps
$
 TOCC(1 SOUR K21 1): NJ = "NEZ", NINJ = "NEZ"
$
$ A table of times cooresponding to the BLOCKS must be generated in
$ SOUR TIME.
$ For SOUR TIME to work it must include a time entry for TIMI and TIMF
$ because the processor has to be able to interpolate needed times; it
$ will not perform extrapolations.
$
$ ================================================================
```

```
$
$ End of Segment MODEL
$
$ ================================================================
$
$ ----------------------------------------------------------------
$
$  Section NODE - defines the nodal positions
$
$ ----------------------------------------------------------------
$
*XQT U1
$
*(29 GENE NODE) NODE
*XQT TAB
START "NT"      $ Define the total number of nodes
JLOC            $ Give the location of the nodes (set up the mesh)
$
$   In the next statement, FORMAT=2 is used for cylindrical coordinates;
$   N1 is the number to start the node locations at (in this case, 1);
$   see page 3.1-7 (green) in the EAL manual for notation
$
FORMAT = 2
$
!XCT=0
!LXCT=LR
*LABEL 501
!XCT = XCT + 1
!LXCT = -2.0*FLOAT(XCT-1)*DX + LXCT
$
!ZCT = 0
*LABEL 502
!ZCT = ZCT + 1
$
!NN = (XCT-1)*NNZ + ZCT
!LZCT = (ZCT-1) * LEZ
$
"NN", "LXCT", 0., "LZCT"
$
*IF ( "ZCT" LT "NNZ" ): *GOTO 502
*IF ( "XCT" LT "NNX" ): *GOTO 501
$
$ "N1", 0., 0., 0., 0., "LZ", 0., "NNZ", 1,"NNX"
$           "NNZ", "LX", 0. 0., "LX", "LZ", 0.
$
*RETURN
*NODE
$
*DCALL(29 GENE NODE)
$
$ ----------------------------------------------------------------
$
$  Section ATTR - Set Element Attributes
$
$ ----------------------------------------------------------------
$
*XQT AUS
$
 TABLE(NI=1,NJ=1): K AREA: J=1: 1.    $The area of K21 elements
 TABLE(NI=1,NJ=1): K THIC: J=1: 1.    $Thickness of K41 elements
$
$ ----------------------------------------------------------------
$
$  Section ELEM - defines the element connectivity
$
$ ----------------------------------------------------------------
$
*XQT ELD           $See page 3.2-1 (green) for explaination
RESET NUTED=1      $Store all TED Kij datasets in Library 1
$
$  Define K41 elements
$  K41 signifies a conductive, 4 node element.
$
 K41
 GROUP = 1         $Group 1 (the only group of K41's in this case)
 NMAT=1            $Material 1
!J1=1              $J1 is the number of the first node in an element
!J2=J1+1           $J2 is the number of the next. NOTE: the nodes are
!J3=1+NNZ+1
!J4=J3-1
 "J1" "J2" "J3" "J4", 1, "NEZ", "NEX"
$
$  Define K21 elements
$  K21 is used for the "source" flux on the exterior so there are not
$  any for the specified temperature case.
$
 K21
 GROUP = 1         $Group 1 (the only group of K21's in this case)
 NMAT = 2          $Material 2 (since it is different from the solid)
!J1=1             $Nodes on outside surface
```

```
!J2=J1+1
 "J1" "J2", 1, "NEZ"
$
$ Define K21 elements for the line source heat flux boundary (above).
$ Note: since we're using this element to model the heat flux, the
$ thermal conductivities and specific heat must be zero.
$
$
$ ----------------------------------------------------------------
$
$  Section TGEO - put it all together
$
$ ----------------------------------------------------------------
$
*XQT TGEO
*XQT DCU
 XCOPY 1,6 JLOC BTAB
$
$ ================================================================
$
$ End of Segment GEOMETRY
$
$ ================================================================
$
```


E.3 R58 Module

```
$ ================================================================
$
$ Segment GEOMETRY define geometry for 2-D shell
$                  (one conduction layer)
$
$ ================================================================


!LM=0.00635           $Radius (thickness) of the model (m)
!LG=0.0000508         $thickness of glue layer (2 mils in m)
!LU=0.0000508         $thickness of upilex (2 mils in m)

!RU = LM +LG +LU      $total model thickness
!RG = LM +LG

!F=1.0              $Integration scheme

!PI = ACOS(-1.0)


$ ================================================================
$
$ Segment MODEL define properties, input data, initial values
$ modl.r14.seg
$
$ ================================================================

$ ----------------------------------------------------------------
$
$  Section PROP - thermal property data
$
$ ----------------------------------------------------------------

*XQT AUS

 DEFINE C =  2 VARB PROP

$ Read in the varaible property data (T, rho, c, kxx)
$ NVE, the number of property entries must be defined in the data file
*TF OPEN 1'varbprop1
*TF READ 1
*TF CLOSE 1
 TABLE(NI=9,NJ="NVE"): COND PROP 1
 TRANSFER(SOURCE=C,DSKIP=3,ILIM=6,JLIM="NVE",OPER=XSUM)

*TF OPEN 2'varbprop2
*TF READ 2
*TF CLOSE 2
 TABLE(NI=9,NJ="NVE"): COND PROP 2
 TRANSFER(SOURCE=C,DSKIP=3,ILIM=6,JLIM="NVE",OPER=XSUM)

*TF OPEN 3'varbprop3
*TF READ 3
*TF CLOSE 3
 TABLE(NI=9,NJ="NVE"): COND PROP 3
 TRANSFER(SOURCE=C,DSKIP=3,ILIM=6,JLIM="NVE",OPER=XSUM)

$ NI=9 indicates that nine variables can be entered to determine k
$ (T, rho, c, kxx, kyy, kzz, kxy, kzy, kzx).

$ For COND PROP n, n is the material number
$ For SOUR K21 n, n is the group number
```

```
$ The table of properties read in contain (T, rho, c, kxx). These
$ are being transferred to COND PROP

!DELT = 0.0002
!DELT

$ Thermal property table for the flux source.

 TABLE(NI=9,NJ=1): COND PROP 4

*XQT DCU
 PRINT 1 COND PROP 1
 PRINT 1 COND PROP 2
 PRINT 1 COND PROP 3
 PRINT 1 COND PROP 4

$ ----------------------------------------------------------------
$
$  Section SLOC - from sensor location, get nodes for sensors
$
$ ----------------------------------------------------------------

*XQT U1

*(29 MK SLOC) SLOC

*XQTC AUS

TABLE(NI=1,NJ="NS",TYPE=0): MINT NODE

!SCT = 0
*LABEL 621
    !SCT = SCT + 1

    !CSL = DS 1, "SCT", 1 (SENS LOCS)
    !SNN = IFIX((CSL - SL0) /(SLN - SL0) *FLOAT(NEZ)) +1

    TABLE,U(TYPE=0): MINT NODE: I=1: J="SCT": "SNN"

    !ENDS = SCT - NS
*JLZ(ENDS, 621)

DE1: SOURCE=MINT NODE: DEST=MINT NODE 2: EX1

!CORR = DS 1, "NS", 1 (MINT NODE)
!CORR = CORR - 1
TABLE,U(TYPE=-2): MINT NODE 2: J="NS": "CORR"

*RETURN
*SLOC

$ ----------------------------------------------------------------
$
$  Section INPT - determine the measured temperature data
$                 initialize data sets
$
$ ----------------------------------------------------------------

*XQTC AUS

!NR = 1       $number of time steps to be considered at once

$ Note that NTS is in sourtime and NS is in senslocs
$ they must be read before meastemp

*TF OPEN 4'sourtime.eal
*TF READ 4
*TF CLOSE 4

*TF OPEN 7'senslocs.eal
*TF READ 7
*TF CLOSE 7

*TF OPEN 8'meastemp.eal
*TF READ 8
*TF CLOSE 8

$ Based on the input read, we can define the rest of the parms

!NVS = NS +2    $number of virtual nodes used by lin5

!SL0 = DS 1, 1, 1 (1 SENS LOCS)
!SLN = DS 1, "NS", 1 (1 SENS LOCS)
!LZ = SLN - SL0

$ the formula for NEZ was developed through trial and error
$ so that the appropriate number of nodes exists per run
$ the rigging at the end makes sure NNZ is odd and
$ not too small
```

```
!NEZ = IFIX(LZ *10000) /2 *2 +2
!DZ = LZ /NEZ       $size of element
!NNZ=NEZ +1         $Number of nodes
!DZ
!NEZ

!NEU=8              $Number of elements
!DU = LU /NEU       $size of element
!NNU=NEU +1         $Number of nodes
!DU
!NEU

!NEG=5              $Number of elements
!DG = LG /NEG       $size of element
!NNG=NEG +1         $Number of nodes
!DG
!NEG

!NEM=10             $Number of elements
!DM = LM /NEM       $size of element
!NNM=NEM +1         $Number of nodes
!DM
!NEM

!NT= NNM +NEU +NEG *NNZ  $Total number of nodes
!NT
!N1=1                 $Beginning node for exterior
!N2=NEU *NNZ +1       $Beginning node for layer 2
!N3=NEU +NEG *NNZ +1  $Beginning node for layer 3

!NTI=NS*NR

*DCALL (29 MK SLOC)

 TABLE(NI="NTI",NJ=1): Y
 TABLE(NI="NTI",NJ=1): T
 TABLE(NI="NTI",NJ=1): TDQ
$
!TEMI = DS 1, 1, 1 ("HL" MEAS TEMP)
 TABLE(NI=1,NJ="NT"): "HL" TRAN TEMP 2: J=1,"NT": "TEMI"
 TABLE(NI=1,NJ="NT"): "HL" INIT TEMP
 TABLE(NI="NNZ",NJ="NTS"): "HL" SURF TEMP
$
 DEFINE TT2 = "HL" TRAN TEMP 2
!S1 = N2 - 1
 TABLE,U(TYPE=-2): "HL" SURF TEMP
 TRANSFER(SOURCE=TT2,SBASE="S1",ILIM="NNZ",OPER=XSUM)

$This represents HTH for NS=5 two virtual nodes (7x10)
!NHI = NS *2
TABLE(NI="NHI",NJ="NVS"): HFT5
  J=1:  -1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0
  J=2:  -1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0
  J=3:  0.0 -1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0
  J=4:  0.0 0.0 -1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0
  J=5:  0.0 0.0 0.0 -1.0 0.0 0.0 0.0 0.0 1.0 0.0
  J=6:  0.0 0.0 0.0 0.0 -1.0 0.0 0.0 0.0 0.0 1.0
  J=7:  0.0 0.0 0.0 0.0 -1.0 0.0 0.0 0.0 0.0 1.0
TABLE(NI="NVS",NJ="NVS"): HDT5
  J=1: 1.0 0.0 0.0 0.0 0.0 0.0 0.0
  J=2: 0.0 1.0 0.0 0.0 0.0 0.0 0.0
  J=3: 0.0 0.0 1.0 0.0 0.0 0.0 0.0
  J=4: 0.0 0.0 0.0 1.0 0.0 0.0 0.0
  J=5: 0.0 0.0 0.0 0.0 1.0 0.0 0.0
  J=6: 0.0 0.0 0.0 0.0 0.0 1.0 0.0
  J=7: 0.0 0.0 0.0 0.0 0.0 0.0 1.0
TABLE(NI="NHI", NJ="NVS"): HFS5
  J=1: 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
  J=2: 0.0 0.0 0.0 0.0 0.0 1.0 -1.0 0.0 0.0 0.0
  J=3: 0.0 0.0 0.0 0.0 0.0 -1.0 2.0 -1.0 0.0 0.0
  J=4: 0.0 0.0 0.0 0.0 0.0 0.0 -1.0 2.0 -1.0 0.0
  J=5: 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -1.0 2.0 -1.0
  J=6: 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -1.0 1.0
  J=7: 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
TABLE(NI="NVS", NJ="NVS"): HDS5
  J=1: 1.0 -1.0 0.0 0.0 0.0 0.0 0.0
  J=2: -1.0 2.0 -1.0 0.0 0.0 0.0 0.0
  J=3: 0.0 -1.0 2.0 -1.0 0.0 0.0 0.0
  J=4: 0.0 0.0 -1.0 2.0 -1.0 0.0 0.0
  J=5: 0.0 0.0 0.0 -1.0 2.0 -1.0 0.0
  J=6: 0.0 0.0 0.0 0.0 -1.0 2.0 -1.0
  J=7: 0.0 0.0 0.0 0.0 0.0 -1.0 1.0

!ALPT = 0.0
!ALPS = 0.0
RFT5 = RTRAN( "ALPT" HFT5 )
RDT5 = RTRAN( "ALPT" HDT5 )
RFS5 = RTRAN( "ALPS" HFS5 )
RDS5 = RTRAN( "ALPS" HDS5 )
```

```
*XQT DCU
 PRINT 1 RF5
 PRINT 1 MINT NODE
 PRINT "HL" MEAS TEMP
 PRINT 1 SOUR TIME
 XCOPY "HL",7 MEAS TEMP
 XCOPY 1, 9 MINT NODE

$ ----------------------------------------------------------------
$
$  Section BOUN - initialize boundary conditions
$
$ ----------------------------------------------------------------

*XQT AUS

 TABLE(NI="NEZ",NJ="NTS"): SOUR K21 1

$ The table of fluxes just created must be put into the form such that
$ J is the number of elements that have a source value
$ BLOCK is the number of time steps

 TOCC(1 SOUR K21 1): NJ = "NEZ", NINJ = "NEZ"

$ A table of times cooresponding to the BLOCKS must be generated in
$ SOUR TIME.
$ For SOUR TIME to work it must include a time entry for TIMI and TIMF
$ because the processor has to be able to interpolate needed times; it
$ will not perform extrapolations.

$ ----------------------------------------------------------------
$
$  Section FLUX DIST - update flux estimate
$
$ ----------------------------------------------------------------

*XQT U1

*(29 FLUX DIST) DIST

*XQTC AUS

!NSCT = 0
!NDI = NS +2
TABLE(NI="NDI", NJ="NEZ"): FLUX DIST

*LABEL 992 $loop over sensors
    !NSCT = NSCT +1

$ Get node number of Middle, Left and Right sensors
    !JM = NSCT -1
    !JL = JM -2
    !JR = JM +2

    !NSL = 0
    !NSR = NNZ +1

    !NSMC = 1
*   IF ( "JM" LT 1 ): !NSMC = 0: !NSM = 0
*   IF ( "JM" GT "NS" ): !NSMC = 0: !NSM = NNZ +1
*   IF ( "NSMC" EQ 1 ): !NSM = DS "JM", 1, 1 (MINT NODE)

*   IF ( "JL" GE 1 ): !NSL = DS "JL", 1, 1 (MINT NODE)
*   IF ( "JR" LE "NS" ): !NSR = DS "JR", 1, 1 (MINT NODE)

    !JCT = NSL
*   IF ( "JCT" LT 1 ): !JCT = 1

*   LABEL 997   $ loop over nodes around sensor

*       IF ( "JCT" LT "NSM" ): !QF = FLOAT(JCT -NSL) +0.5 /FLOAT(NSM -NSL)
*       IF ( "JCT" GE "NSM" ): !QF = FLOAT(NSR -JCT) -0.5 /FLOAT(NSR -NSM)

$ put QF in FLUX DIST for triangular patch
$ and put 1.0 for constant patch

 TABLE,U(TYPE=-2): 1 FLUX DIST: I="NSCT": J="JCT": "QF"

        !JCT = JCT + 1
        !ENDJ = JCT - NSR
*       IF ( "JCT" GT "NEZ" ): !ENDJ = 1
*    JLZ(ENDJ, 997)

     !ENDI = NSCT -NS -2
*JLZ(ENDI,992)


*RETURN
```

```
*DIST

*DCALL(29 FLUX DIST)

*XQT DCU
 PRINT 1 FLUX DIST

$ ================================================================
$
$ End of Segment MODEL
$
$ ================================================================
$
$ ----------------------------------------------------------------
$
$  Section NODE - defines the nodal positions
$
$ ----------------------------------------------------------------

*XQT U1
$
*(29 GENE NODE) NODE
*XQT TAB
START "NT"      $ Define the total number of nodes
JLOC            $ Give the location of the nodes (set up the mesh)

$   In the next statement, FORMAT=2 is used for cylindrical coordinates;
$   N1 is the number to start the node locations at (in this case, 1);
$   see page 3.1-7 (green) in the EAL manual for notation

FORMAT = 2

$ define the upilex layer
!RBU = RG +DU
"N1", "RU", 0.0, 0.0, "RU", 0.0, "LZ", "NNZ", 1, "NEU"
"NNZ", "RBU", 0.0, 0.0, "RBU", 0.0, "LZ"

$ define the glue layer
!RBG = LM +DG
"N2", "RG", 0.0, 0.0, "RG", 0.0, "LZ", "NNZ", 1, "NEG"
"NNZ", "RBG", 0.0, 0.0, "RBG", 0.0, "LZ"

$define the macor layer with a geometric grid

!XCT=0
!LXCT=LM
*LABEL 501
    !XCT = XCT +1

    !LXCT = LM *COS(XCT -1 *PI /2 /NNM)
    !NN = XCT -1 *NNZ +N3

    "NN", "LXCT", 0.0, 0.0, "LXCT", 0.0, "LZ", "NNZ", 1

    !ENDX = XCT -NNM
*JLZ(ENDX,501)

*RETURN
*NODE

*DCALL(29 GENE NODE)

*XQT DCU
 XCOPY 1, 3 JLOC BTAB

$ ----------------------------------------------------------------
$
$  Section ATTR - Set Element Attributes
$
$ ----------------------------------------------------------------
$
*XQT AUS
$
 TABLE(NI=1,NJ=1): K AREA: J=1: 1.    $The area of K21 elements
 TABLE(NI=1,NJ=1): K THIC: J=1: 1.    $Thickness of K41 elements
$
$ ----------------------------------------------------------------
$
$  Section ELEM - defines the element connectivity
$
$ ----------------------------------------------------------------
$
*XQT ELD           $See page 3.2-1 (green) for explaination
RESET NUTED=1      $Store all TED Kij datasets in Library 1
$
$  Define K41 elements
$  K41 signifies a conductive, 4 node element.
$
 K41
```

```
 GROUP = 1         $Group 1 (upilex)
 NMAT=1            $Material 1
!J1=N1
!J2=J1+1
!J3=N1+NNZ+1
!J4=J3-1
 "J1" "J2" "J3" "J4", 1, "NEZ", "NEU"

 GROUP = 2         $Group 2 (glue)
 NMAT=2            $Material 2
!J1=N2
!J2=J1+1
!J3=N2+NNZ+1
!J4=J3-1
 "J1" "J2" "J3" "J4", 1, "NEZ", "NEG"

 GROUP = 3         $Group 3 (macor)
 NMAT=3            $Material 3
!J1=N3             $J1 is the number of the first node in an element
!J2=J1+1           $J2 is the number of the next. NOTE: the nodes are
!J3=N3+NNZ+1
!J4=J3-1
 "J1" "J2" "J3" "J4", 1, "NEZ", "NEM"

$  Define K21 elements
$  K21 is used for the "source" flux on the exterior so there are not
$  any for the specified temperature case.

 K21

 GROUP = 1         $Group 1 (the only group of K21's in this case)
 NMAT = 4          $Material 4 (since it is different from the solid)
!J1=N1              $Nodes on outside surface
!J2=J1+1
 "J1" "J2", 1, "NEZ"

$ Define K21 elements for the line source heat flux boundary (above).
$ Note: since we're using this element to model the heat flux, the
$ thermal conductivities and specific heat must be zero.


$ ------------------------------------------------------------------
$
$  Section TGEO - put it all together
$
$ ------------------------------------------------------------------
$
*XQT TGEO
*XQT DCU
 XCOPY 1,6 JLOC BTAB
$
$ ==================================================================
$
$ End of Segment GEOMETRY
$
$ ==================================================================
$
```

E.4 Direct Calculation Module

```
$ ==================================================================
$
$ Segment CALCULATE perform the analysis (direct)
$ calc.dir.seg
$
$ ==================================================================

*XQT U1

$ ------------------------------------------------------------------
$
$  Section DRCT - direct conduction solver
$
$ ------------------------------------------------------------------

*(29 DRCT SLTN) DRCT

*XQT TRTB            $Transient analysis processor
 RESET T1="TSTR"
 RESET T2="TSTP"
 RESET DT="DELT"
 RESET KTI=1
 RESET DEST="TL"
 RESET PRINT=0
 RESET MXNDT=100000
 RESET CORE = 6291456  $6 MEG
```

```
 KTIME="DELK"
 TEMP = "TEMI"
 TSAVE=1 SOUR TIME

*RETURN
*DRCT

*XQT AUS

!TSTR = DS 1, 1, 1 (SOUR TIME)
!TSTP = DS 1, "NTS", 1 (SOUR TIME)

$ ----------------------------------------------------------------
$
$  Section SLTN - run the solution procedure
$
$ ----------------------------------------------------------------

*DCALL (29 DRCT SLTN)

*XQT AUS
 DEFINE TT2 = "TL" TRAN TEMP
!SS = NT - NNZ
 TABLE(NI="NNZ",NJ="NTS"): "HL" SURF TEMP
 TRANSFER(SOURCE=TT2,SSKIP="SS",ILIM="NNZ",OPER=XSUM)

TABLE(NI="NS",NJ="NTS"): T
DE1: SOURCE="HL" SURF TEMP 1 1 1: DEST,U = T
 IS=MINT NODE: EX1

*XQT DCU
 XCOPY 1, 8 SOUR K21
 XCOPY "TL", 10 TRAN TEMP
 XCOPY 1, 11 T
 PRINT 1 T
```

E.5 Function Specification Module

```
$ ================================================================
$
$ Segment CALCULATE perform the analysis
$
$ ================================================================
*XQT U1

$ ----------------------------------------------------------------
$
$  Section TRES - calculate temperature residual
$
$ ----------------------------------------------------------------

*(29 CALC TRES) TRES
*XQT AUS
$
!RCTR=0
*LABEL 951
    !RCTR=RCTR+1

    !RBLK=TCTR+RCTR
    !ROFF=RCTR -1 *NS

    DE1: SOURCE="HL" MEAS TEMP 1 1 1: DEST,U = Y
     JS="RBLK": IDBASE="ROFF": EX1
    DE1: SOURCE="HL" SURF TEMP 1 1 1: DEST,U = T
     JS="RBLK": IDBASE="ROFF": IS=MINT NODE: EX1

    !ENDR=RCTR-NR
*JLZ(ENDR,951)

*RETURN
*TRES

$ ----------------------------------------------------------------
$
$  Section DRCT - direct conduction solver
$
$ ----------------------------------------------------------------

*(29 DRCT SLTN) DRCT

*XQT TRTB            $Transient analysis processor
 RESET T1="TSTR"
 RESET T2="TSTP"
 RESET DT="DELT"
 RESET KTI=1
 RESET DEST="TL"
 RESET PRINT=0
```

```
 KTIME="DELK"
 TEMP = "HL" INIT TEMP 1 1
 TSAVE=1 SOUR TIME

*XQT AUS

!TBLK=TCTR+1
 DEFINE TT = "TL" TRAN TEMP 1 1 2 2
 TABLE,U(TYPE=-2): "HL" TRAN TEMP 2
 TRANSFER(SOURCE=TT,OPER=XSUM)

!S1 = N2 - 1
!S2 = TCTR -1 * NNZ
 DEFINE TT = "TL" TRAN TEMP 1 1
 TABLE,U(TYPE=-2): "HL" SURF TEMP
 TRANSFER(SOURCE=TT,SBASE="S1",DBASE="S2",ILIM="NNZ",OPER=XSUM)

*RETURN
*DRCT

$ ----------------------------------------------------------------
$
$  Section INVS - main sequential time loop
$
$ ----------------------------------------------------------------

*(29 INVS SLTN) INVS
*XQTC AUS

!TCTR = 0
*LABEL 901       $Main time loop

    !TCTR = TCTR + 1
    !TCTR
    !TS2 = TCTR + NR
    !TSTR = DS 1, "TCTR", 1 (SOUR TIME)
    !TSTP = DS 1, "TS2", 1 (SOUR TIME)

    DEFINE TT2 = "HL" TRAN TEMP 2
    TABLE,U(TYPE=-2):"HL" INIT TEMP
    TRANSFER(SOURCE=TT2,OPER=XSUM)

$ Calculate k-matrix refactor time based on temperature change
    !YPRV = DS 3, "TCTR", 1 ("HL" MEAS TEMP)
    !YCRR = DS 3, "TS2", 1 ("HL" MEAS TEMP)
    !YDFF = IFIX ( ABS( YCRR -YPRV ) /3.0 )
*   IF ("YDFF" LT 2): !YDFF = 2
    !DELK = 0.002 *FLOAT( NR ) /FLOAT( YDFF )
!DELK

    !QGSS=TS2 -1
    !NCT=0
*   LABEL 633  $ make estamate from previous time be initial guess

 !NCT = NCT + 1
 !QV = DS 1, "NCT", "QGSS" (1 SOUR K21 1)
 TABLE,U(TYPE=-2): SOUR K21 1: OPER=XSUM: BLOCK "TS2": J="NCT": "QV"
 !ENDN = NCT - NEZ
*   JLZ(ENDN,633)

    !ITER=0
*   LABEL 630        $Iteration loop

 !ITER=ITER+1
!ITER
*  DCALL(29 CALC TRES) $Calculates temperature vectors
*  DCALL(29 DRCT SLTN) $Direct Solution
*  DCALL(29 SENS COEF) $Sensitivity coefficient matrix

*  XQTC AUS
         !TMP = TCTR +1
         DE1: SOURCE = 1 SOUR K21 1 1 "TCTR", "TMP"
              DEST = "TL" Q0
              JS = 1 MINT NODE 2
              EX1
         TOCC("TL" Q0): NJ=1, NINJ="NHI"
 OUTLIB=13
 INLIB=13
 DEFINE Z = "TL" X
 DEFINE Y = 1 Y
 DEFINE T = 1 T
         DEFINE RFT5 = 1 RFT5
         DEFINE RDT5 = 1 RDT5
         DEFINE RFS5 = 1 RFS5
         DEFINE RDS5 = 1 RDS5

$ Calc (XTX + HTH)dq = XT(Y-T) - HTHq
$    HTH is RD5 on the LHS and RF5 on the right hand side.
 XT = RTRAN(Z)
 YMT = SUM(Y,"CY" T)
```

```
 XTX = RPROD(XT,Z)
         RD5 = SUM( RDT5, RDS5 )
         LHS = SUM(XTX, RD5)
 XTYMT = RPROD( XT, YMT )
         HHQT = RPROD(RFT5, Q0)
         HHQS = RPROD(RFS5, Q0)
         HHQ = SUM( HHQT, HHQS )
         RHS = SUM(XTYMT, "CY" HHQ)
 LHSI = RINV( LHS )
 DQ = RPROD( LHSI, RHS )

 MMDQ = MM1( DQ )
 !MXDQ = DS 3,1,1 ( "TL" MMDQ )

*  XQTC AUS

 OUTLIB=1
 INLIB=1

*        DCALL(29 UPDT FLX2) $Update flux estimate
!IPUSS = DS 8, 1, 1 ("TL" Q0 )
!IPUSS
!IDICK = DS 3, 1, 1 ("TL" Q0 )
!IDICK

*        XQT DCU
$PRINT "TL" Q0
$PRINT "HL" INIT TEMP
$PRINT "HL" TRAN TEMP 2
$PRINT 1 Y
$PRINT 1 T
$PRINT "TL" YMT
$PRINT "TL" X
$PRINT "TL" XTX
$PRINT "TL" XTYMT
$PRINT "TL" LHSI
$PRINT "TL" RHS
$PRINT "TL" HHQ
PRINT "TL" DQ
$PRINT 1 SOUR K21
         ERASE "TL"
$*XQT EXIT
*  IF ("MXDQ" LT 1000.0): *GOTO 610
*  IF ("ITER" GT 51): *GOTO 610
*  GOTO 630

*   LABEL 610           $End iteration loop

!PUSS = DS 1, 10, "TCTR" ( 1 SOUR K21 )
!PUSS
!DICK = DS 1, 10, 1 ( "HL" INIT TEMP )
!DICK

*   XQTC AUS
    DEFINE TT = "HL" TRAN TEMP 2
    TABLE,U(TYPE=-2): "HL" INIT TEMP
    TRANSFER(SOURCE=TT,OPER=XSUM)

    !JTST = TS2 - NTS
*JLZ (JTST, 901)

*RETURN
*INVS

$ ---------------------------------------------------------------
$
$  Section SLTN - run the solution procedure
$
$ ---------------------------------------------------------------

*DCALL(29 INVS SLTN) $Direct Transient Solver

*XQT DCU
 XCOPY 1, 8 SOUR K21
```

E.6 Regularization Module

```
$ ================================================================
$
$ Segment CALCULATE perform the analysis
$
$ ================================================================

*XQT U1

$ ----------------------------------------------------------------
$
$  Section TRES - calculate temperature residual
$
$ ----------------------------------------------------------------

*(29 CALC TRES) TRES
*XQT AUS

$ Get surface temperatures
!S2 = TCTR  *NNZ
DEFINE TT = "HL" TRAN TEMP 2 1
TABLE,U(TYPE=-2): "HL" SURF TEMP
  TRANSFER(SOURCE=TT,DBASE="S2",ILIM="NNZ",OPER=XSUM)

$ Get surface temperatures at certain nodes
!RCTR = 0
*LABEL 951
    !RCTR = RCTR +1

    !RBLK = TCTR +RCTR
    !ROFF = RCTR -1 *NS

    DE1: SOURCE="HL" MEAS TEMP 1 1 1: DEST,U = Y
     JS="RBLK": IDBASE="ROFF": EX1
    DE1: SOURCE="HL" SURF TEMP: DEST,U = T
     JS="RBLK": IDBASE="ROFF": IS=MINT NODE: EX1

    !ENDR = RCTR -NR
*JLZ(ENDR,951)

*RETURN
*TRES

$ ----------------------------------------------------------------
$
$  Section DRCT - direct conduction solver
$
$ ----------------------------------------------------------------

*(29 DRCT SLTN) DRCT

*XQT TRTB            $Transient analysis processor
 RESET T1="TSTR"
 RESET T2="TSTP"
 RESET DT="DELT"
 RESET KTI=1
 RESET DEST="TL"
 RESET PRINT=0
 KTIME="DELK"
 TEMP = "HL" INIT TEMP 1 1
 TSAVE=1 SOUR TIME

*XQT AUS

!TBLK=TCTR+1
 DEFINE TT = "TL" TRAN TEMP 1 1 2
 TABLE,U(TYPE=-2): "HL" TRAN TEMP 2
 TRANSFER(SOURCE=TT,OPER=XSUM)

*RETURN
*DRCT

$ ----------------------------------------------------------------
$
$  Section INVS - main sequential time loop
$
$ ----------------------------------------------------------------

*(29 INVS SLTN) INVS
*XQTC AUS

!TCTR = 0
*LABEL 901       $Main time loop

$ Define time step parameters
    !TCTR = TCTR + 1
    !TCTR
    !TS2 = TCTR + NR
```

```
    !TSTR = DS 1, "TCTR", 1 (SOUR TIME)
    !TSTP = DS 1, "TS2", 1 (SOUR TIME)

$ Establish initial temperature for this time step
    DEFINE TT2 = "HL" TRAN TEMP 2 1 1
    TABLE,U(TYPE=-2):"HL" INIT TEMP
    TRANSFER(SOURCE=TT2,OPER=XSUM)

$ Calculate k-matrix refactor time based on temperature change
    !YPRV = DS 3, "TCTR", 1 ("HL" MEAS TEMP)
    !YCRR = DS 3, "TS2", 1 ("HL" MEAS TEMP)
    !YDFF = IFIX ( ABS( YCRR -YPRV ) /3.0 )
*   IF ("YDFF" LT 2): !YDFF = 2
    !DELK = 0.002 *FLOAT( NR ) /FLOAT( YDFF )

$ make estimate from previous time be initial guess
    !QGSS = TS2 -1
    !NCT = 0
*   LABEL 633
 !NCT = NCT + 1
 !QV = DS 1, "NCT", "QGSS" (1 SOUR K21 1)
 TABLE,U(TYPE=-2): SOUR K21 1: OPER=XSUM: BLOCK "TS2": J="NCT": "QV"
 !ENDN = NCT -NEZ
*   JLZ(ENDN,633)

    !ITER=0
*   LABEL 630        $Iteration loop

 !ITER=ITER+1
!ITER

*  DCALL(29 CALC TRES) $Calculates temperature vectors
*  DCALL(29 SENS COEF) $Sensitivity coefficient matrix

$ Get current heat flux estimates
*  XQTC AUS
         !TMP = TCTR +1
         DE1: SOURCE = 1 SOUR K21 1 1 "TCTR", "TMP"
              DEST = "TL" Q0
              JS = 1 MINT NODE 2
              EX1
         TOCC("TL" Q0): NJ=1, NINJ="NHI"

$ Set parms
 OUTLIB=13
 INLIB=13
 DEFINE Z = "TL" X
 DEFINE Y = 1 Y
 DEFINE T = 1 T
         DEFINE RFT5 = 1 RFT5
         DEFINE RDT5 = 1 RDT5
         DEFINE RFS5 = 1 RFS5
         DEFINE RDS5 = 1 RDS5

$ Calc (XTX + HTH)dq = XT(Y-T) - HTHq
$    HTH is RD5 on the LHS and RF5 on the right hand side.
 XT = RTRAN(Z)
 YMT = SUM(Y,"CY" T)
 XTX = RPROD(XT,Z)
         RD5 = SUM( RDT5, RDS5 )
         LHS = SUM(XTX, RD5)
 XTYMT = RPROD( XT, YMT )
         HHQT = RPROD(RFT5, Q0)
         HHQS = RPROD(RFS5, Q0)
         HHQ = SUM( HHQT, HHQS )
         RHS = SUM(XTYMT, "CY" HHQ)
 LHSI = RINV( LHS )
 DQ = RPROD( LHSI, RHS )

$ Test for convergence
 MMDQ = MM1( DQ )
 !MXDQ = DS 3,1,1 ( "TL" MMDQ )

*        DCALL(29 UPDT FLX2) $Update flux estimate
*        DCALL(29 DRCT SLTN) $Calculate direct solution

*  XQTC AUS

$ Debugging section
!IPUSS = DS 6, 1, 1 ("TL" Q0 )
!IPUSS
!IDICK = DS 2, 1, 1 ("TL" Q0 )
!IDICK

*XQT DCU
$PRINT "HL" TRAN TEMP 2 1 1,"NNZ" 1,1 1,1
$PRINT "TL" Q0
$PRINT "HL" INIT TEMP
$PRINT "HL" TRAN TEMP 2
$PRINT 1 Y
```

```
$PRINT 1 T
$PRINT "TL" YMT
PRINT "TL" X
$PRINT "TL" XTX
$PRINT "TL" XTYMT
$PRINT "TL" LHSI
$PRINT "TL" RHS
$PRINT "TL" HHQ
PRINT "TL" DQ
$PRINT 1 SOUR K21
$*XQT EXIT


$ Clean up
*        XQT AUS
 OUTLIB=1
 INLIB=1
*        XQT DCU
         ERASE "TL"

*  IF ("MXDQ" LT 0.1): *GOTO 610
*  IF ("ITER" GT 51): *GOTO 610

*  GOTO 630

*   LABEL 610           $End iteration loop

!PUSS = DS 1, 10, "TCTR" ( 1 SOUR K21 )
!PUSS
!DICK = DS 1, 10, 1 ( "HL" INIT TEMP )
!DICK

*   XQTC AUS
    DEFINE TT = "HL" TRAN TEMP 2
    TABLE,U(TYPE=-2): "HL" INIT TEMP
    TRANSFER(SOURCE=TT,OPER=XSUM)

    !JTST = TS2 - NTS
*JLZ (JTST, 901)

*RETURN
*INVS

$ ---------------------------------------------------------------
$
```



```
$  Section SLTN - run the solution procedure
$
$ ---------------------------------------------------------------

*DCALL(29 INVS SLTN) $Direct Transient Solver

*XQT DCU
 XCOPY 1, 8 SOUR K21
```

E.7 Constant Sensitivity 3-Point Module

```
$ ===============================================================
$
$ Segment FLUX perform flux manipulations
$ File flux.lin.seg
$
$ ===============================================================

*XQT U1

$ ---------------------------------------------------------------
$
$  Section SENS - calculate sensitivity coefficients
$
$ ---------------------------------------------------------------

*(29 SENS COEF) SENS

*XQT AUS

!NXI=NS*NR
```

```
!NXJ=NS
 TABLE(NI="NXI",NJ="NXJ"): "TL" X

!NSCT = 0
*LABEL 902 $loop over sensors
    !NSCT = NSCT + 1

    !SENS = 1.0
*   DCALL (29 UPDT FLUX)

*   XQT TRTB            $Transient analysis processor
    RESET T1="TSTR"
    RESET T2="TSTP"
    RESET DT="DELT"
    RESET KTI=1
    RESET DEST="TL"
    RESET PRINT=0
    KTIME="DELK"
    TEMP = "HL" INIT TEMP 1 1
    TSAVE=1 SOUR TIME

*   XQT AUS
    !SENS = -1.0
*   DCALL (29 UPDT FLUX)

*   XQT AUS
    TOCC("TL" TRAN TEMP): NJ=1, NINJ="NT"
    !FCTR = 0
*   LABEL 961

 !FCTR = FCTR + 1
 !FBLK = FCTR + 1
 !TOFF=NS*(FCTR-1)

 DE1: SOURCE="TL" TRAN TEMP 1 1 "FBLK": DEST,U = TDQ
      IDBASE="TOFF": IS=MINT NODE: EX1

 !ENDF=FCTR-NR
*   JLZ(ENDF,961)

    OUTLIB = "TL"
    XI = SUM(TDQ,"CY" T)
    OUTLIB = 1
    DE1: SOURCE="TL" XI: DEST,U = "TL" X: JD="NSCT": EX1

$*XQT DCU
$PRINT 1 TDQ
$PRINT "TL" XI

    !ENDJ=NSCT - NS
*JLZ(ENDJ,902)

*RETURN
*SENS

$ ---------------------------------------------------------------
$
$  Section FLUX - update flux estimate
$
$ ---------------------------------------------------------------

*(29 UPDT FLUX) FLUX

*XQT AUS

$ Get node number of Middle, Left and Right sensors
!JM = NSCT
!JL = NSCT -1
!JR = NSCT +1
!NSM = DS "JM", 1, 1 (MINT NODE)
!NSL = 1
!NSR = DS "NS", 1, 1 (MINT NODE)
*IF ( "JL" GT 0 ): !NSL = DS "JL", 1, 1 (MINT NODE)
*IF ( "JR" LT "NS" ): !NSR = DS "JR", 1, 1 (MINT NODE)

!NBEG = NSM -IFIX (NSM -NSL /2)
!NEND = NSM +IFIX (NSR -NSM /2) -1

!QF = SENS
*IF ( "SENS" EQ 0.0 ): !QF = DS "NSCT", 1, 1 ("TL" DQ)

!BCT = TCTR
*LABEL 928
    !BCT = BCT +1
    !QF3 = BCT -TCTR *QF

    TABLE,U(TYPE=-1): SOUR K21 1: BLOCK "BCT": J="NBEG","NEND": "QF3"
    !ENDQ = BCT -TCTR -NR
*JLZ(ENDQ, 928)
```

```
*RETURN
*FLUX
```

E.8 Constant Sensitivity 5-Point Module

```
$ ================================================================
$
$ Segment FLUX perform flux manipulations
$ File flux.lin.seg
$
$ ================================================================

*XQT U1

$ ----------------------------------------------------------------
$
$  Section SENS - calculate sensitivity coefficients
$
$ ----------------------------------------------------------------

*(29 SENS COEF) SENS

*XQT AUS

!NXI=NS*NR
!NXJ=NS
 TABLE(NI="NXI",NJ="NXJ"): "TL" X

!NSCT = 0
*LABEL 902 $loop over sensors
    !NSCT = NSCT + 1

    !SENS = 1.0
*   DCALL (29 UPDT FLUX)

*   XQT TRTB            $Transient analysis processor
    RESET T1="TSTR"
    RESET T2="TSTP"
    RESET DT="DELT"
    RESET KTI=1
    RESET DEST="TL"
    RESET PRINT=0
    KTIME="DELK"
    TEMP = "HL" INIT TEMP 1 1
    TSAVE=1 SOUR TIME

*   XQT AUS
    !SENS = -1.0
*   DCALL (29 UPDT FLUX)

*   XQT AUS
    TOCC("TL" TRAN TEMP): NJ=1, NINJ="NT"
    !FCTR = 0
*   LABEL 961

 !FCTR = FCTR + 1
 !FBLK = FCTR + 1
 !TOFF=NS*(FCTR-1)

 DE1: SOURCE="TL" TRAN TEMP 1 1 "FBLK": DEST,U = TDQ
      IDBASE="TOFF": IS=MINT NODE: EX1

 !ENDF=FCTR-NR
*   JLZ(ENDF,961)

    OUTLIB = "TL"
    XI = SUM(TDQ,"CY" T)
    OUTLIB = 1
    DE1: SOURCE="TL" XI: DEST,U = "TL" X: JD="NSCT": EX1

$*XQT DCU
$PRINT 1 TDQ
$PRINT "TL" XI

    !ENDJ=NSCT - NS
*JLZ(ENDJ,902)

*RETURN
*SENS

$ ----------------------------------------------------------------
$
$  Section FLUX - update flux estimate
$
$ ----------------------------------------------------------------

*(29 UPDT FLUX) FLUX
```

```
*XQT AUS

$ Get node number of Middle, Left and Right sensors
!JM = NSCT
!JL = NSCT -2
!JR = NSCT +2
!NSM = DS "JM", 1, 1 (MINT NODE)
!NSL = 1
!NSR = DS "NS", 1, 1 (MINT NODE)
*IF ( "JL" GT 0 ): !NSL = DS "JL", 1, 1 (MINT NODE)
*IF ( "JR" LT "NS" ): !NSR = DS "JR", 1, 1 (MINT NODE)

!NBEG = NSM -IFIX (NSM -NSL /2)
!NEND = NSM +IFIX (NSR -NSM /2) -1

!QF = SENS
*IF ( "SENS" EQ 0.0 ): !QF = DS "NSCT", 1, 1 ("TL" DQ)

!BCT = TCTR
*LABEL 928
    !BCT = BCT +1
    !QF3 = BCT -TCTR *QF

    TABLE,U(TYPE=-1): SOUR K21 1: BLOCK "BCT": J="NBEG","NEND": "QF3"
    !ENDQ = BCT -TCTR -NR
*JLZ(ENDQ, 928)

*RETURN
*FLUX
```

E.9 Triangular Sensitivity 5-Point Module

```
$ ================================================================
$
$ Segment FLUX perform flux manipulations
$ File flux.lin.seg
$
$ ================================================================

*XQT U1

$ ----------------------------------------------------------------
$
$  Section SENS - calculate sensitivity coefficients
$
$ ----------------------------------------------------------------

*(29 SENS COEF) MKX

*XQTC AUS

!XF = 500.0
!XFN = -1.0 *XF
!XFR = 1.0 /XF
!XFNR = -1.0 /XF

TABLE(NI="NTI",NJ="NVS"): "TL" X

!NSCT = 0
*LABEL 902 $loop over sensors
    !NSCT = NSCT +1

    TABLE(NI="NVS",NJ=1): "TL" DQ
    I="NSCT": J=1: "XF"
*   DCALL (29 UPDT FLX2)

*   XQT TRTB            $Transient analysis processor
    RESET T1="TSTR"
    RESET T2="TSTP"
    RESET DT="DELT"
    RESET KTI=1
    RESET DEST="TL"
    RESET PRINT=0
    KTIME="DELK"
    TEMP = "HL" INIT TEMP 1 1
    TSAVE=1 SOUR TIME

*   XQT AUS

    TABLE(NI="NVS",NJ=1): "TL" DQ
    I="NSCT": J=1: "XFN"
*   DCALL (29 UPDT FLX2)

*   XQTC AUS
    TOCC("TL" TRAN TEMP): NJ=1, NINJ="NT"
    !FCTR = 0
*   LABEL 961
```

```
 !FCTR = FCTR +1
 !FBLK = FCTR +1
 !TOFF = FCTR -1 *NS

 DE1: SOURCE="TL" TRAN TEMP 1 1 "FBLK": DEST,U = TDQ
      IDBASE="TOFF": IS=MINT NODE: EX1

 !ENDF=FCTR -NR
*   JLZ(ENDF,961)

    "TL" XI = SUM("XFR" TDQ,"XFNR" T)
    DE1: SOURCE="TL" XI: DEST,U = "TL" X: JD="NSCT": EX1

    !ENDJ=NSCT -NVS
*JLZ(ENDJ,902)

*RETURN
*MKX

$ ----------------------------------------------------------------
$
$  Section FLUX - update flux estimate
$
$ ----------------------------------------------------------------

*(29 UPDT FLUX) FLUX

*XQTC AUS

$ Get node number of Middle, Left and Right sensors
!JM = NSCT -1
!JL = JM -2
!JR = JM +2

!NSL = 0
!NSR = NNZ +1

!NSMC = 1
*IF ( "JM" LT 1 ): !NSMC = 0: !NSM = 0
*IF ( "JM" GT "NS" ): !NSMC = 0: !NSM = NNZ +1
*IF ( "NSMC" EQ 1 ): !NSM = DS "JM", 1, 1 (MINT NODE)

*IF ( "JL" GE 1 ): !NSL = DS "JL", 1, 1 (MINT NODE)
*IF ( "JR" LE "NS" ): !NSR = DS "JR", 1, 1 (MINT NODE)
```



```
!JCT = NSL
*IF ( "JCT" LT 1 ): !JCT = 1

*LABEL 927   $ loop over nodes around sensor

     !DELQ = SENS
*    IF ( "SENS" EQ 0.0 ): !DELQ = DS "NSCT", 1, 1 ("TL" DQ)

*    IF ( "JCT" LT "NSM" ): !QF = FLOAT(JCT -NSL) +0.5 /FLOAT(NSM -NSL)
*    IF ( "JCT" GE "NSM" ): !QF = FLOAT(NSR -JCT) -0.5 /FLOAT(NSR -NSM)

     !BCT = TCTR
*    LABEL 928
 !BCT = BCT +1
 !QF3 = BCT -TCTR *DELQ *QF  $for triangular patch linear
$  !QF3 = BCT -TCTR *DELQ      $for constant patch linear
$        !QF3 = DELQ *QF             $triangular patch constant
$        !QF3 = DELQ                 $constant patch constant

 TABLE,U(TYPE=-2): 1 SOUR K21 1: BLOCK "BCT": J="JCT": "QF3"
 !ENDQ = BCT -TCTR -NR
*    JLZ(ENDQ, 928)

     !JCT = JCT + 1
     !ENDJ = JCT - NSR

*    IF ( "JCT" GT "NEZ" ): !ENDJ = 1
*JLZ(ENDJ, 927)

*RETURN
*FLUX

$ ----------------------------------------------------------------
$
$  Section UPDT FLX2 - update flux estimate
$
$ ----------------------------------------------------------------

*(29 UPDT FLX2) FLX2

*XQTC AUS

DEFINE SKD = 1 FLUX DIST
DEFINE DQ = "TL" DQ
DEFINE SKU = "TL" SKU
```

```
DEFINE DQT = "TL" DQT

"TL" DQT = RTRAN( DQ )

!BCT = TCTR
*LABEL 555
    !BCT = BCT +1
$    !LINF = FLOAT( BCT -TCTR )
    !LINF = 1.0
    "TL" SKU = RPROD( "LINF" DQT, SKD )

    TABLE,U(TYPE=-2): 1 SOUR K21 1
    TRANSFER(SOUR=SKU, L1="BCT", ILIM=1, JLIM="NEZ")

    !ENDT = BCT -TCTR -NR
*JLZ(ENDT, 555)

*RETURN
*FLX2
```


E.10 Regularization Sensitivity Module

```
$ ================================================================
$
$ Segment FLUX perform flux manipulations
$ File flux.lin.seg
$
$ ================================================================

*XQT U1

$ ----------------------------------------------------------------
$
$  Section SENS - calculate sensitivity coefficients
$
$ ----------------------------------------------------------------

*(29 SENS COEF) MKX

*XQTC AUS

!XF = 0.1
!XFN = -1.0 *XF
!XFR = 1.0 /XF
!XFNR = -1.0 /XF

TABLE(NI="NTI",NJ="NVS"): "TL" X

!NSCT = 0
*LABEL 902 $loop over sensors
    !NSCT = NSCT +1

    TABLE(NI="NVS",NJ=1): "TL" DQ
    I="NSCT": J=1: "XF"
*   DCALL (29 UPDT FLX2)

*   XQT TRTB            $Transient analysis processor
    RESET T1="TSTR"
    RESET T2="TSTP"
    RESET DT="DELT"
    RESET KTI=1
    RESET DEST="TL"
    RESET PRINT=0
    KTIME="DELK"
    TEMP = "HL" INIT TEMP 1 1
    TSAVE=1 SOUR TIME

*   XQT AUS

    TABLE(NI="NVS",NJ=1): "TL" DQ
    I="NSCT": J=1: "XFN"
*   DCALL (29 UPDT FLX2)

*   XQTC AUS
    TOCC("TL" TRAN TEMP): NJ=1, NINJ="NT"
    !FCTR = 0
*   LABEL 961

 !FCTR = FCTR +1
```

```
 !FBLK = FCTR +1
 !TOFF = FCTR -1 *NS

 DE1: SOURCE="TL" TRAN TEMP 1 1 "FBLK": DEST,U = TDQ
      IDBASE="TOFF": IS=MINT NODE: EX1

 !ENDF=FCTR -NR
*   JLZ(ENDF,961)

    "TL" XI = SUM("XFR" TDQ,"XFNR" T)
    DE1: SOURCE="TL" XI: DEST,U = "TL" X: JD="NSCT": EX1

    !ENDJ=NSCT -NVS
*JLZ(ENDJ,902)

*RETURN
*MKX

$ ----------------------------------------------------------------
$
$  Section FLUX - update flux estimate
$
$ ----------------------------------------------------------------

*(29 UPDT FLUX) FLUX

*XQTC AUS

$ Get node number of Middle, Left and Right sensors
!JM = NSCT -1
!JL = JM -2
!JR = JM +2

!NSL = 0
!NSR = NNZ +1

!NSMC = 1
*IF ( "JM" LT 1 ): !NSMC = 0: !NSM = 0
*IF ( "JM" GT "NS" ): !NSMC = 0: !NSM = NNZ +1
*IF ( "NSMC" EQ 1 ): !NSM = DS "JM", 1, 1 (MINT NODE)

*IF ( "JL" GE 1 ): !NSL = DS "JL", 1, 1 (MINT NODE)
*IF ( "JR" LE "NS" ): !NSR = DS "JR", 1, 1 (MINT NODE)

!JCT = NSL
*IF ( "JCT" LT 1 ): !JCT = 1

*LABEL 927   $ loop over nodes around sensor

     !DELQ = SENS
*    IF ( "SENS" EQ 0.0 ): !DELQ = DS "NSCT", 1, 1 ("TL" DQ)

*    IF ( "JCT" LT "NSM" ): !QF = FLOAT(JCT -NSL) +0.5 /FLOAT(NSM -NSL)
*    IF ( "JCT" GE "NSM" ): !QF = FLOAT(NSR -JCT) -0.5 /FLOAT(NSR -NSM)

     !BCT = TCTR
*    LABEL 928
 !BCT = BCT +1
 !QF3 = BCT -TCTR *DELQ *QF  $for triangular patch linear
$  !QF3 = BCT -TCTR *DELQ      $for constant patch linear
$        !QF3 = DELQ *QF             $triangular patch constant
$        !QF3 = DELQ                 $constant patch constant

 TABLE,U(TYPE=-2): 1 SOUR K21 1: BLOCK "BCT": J="JCT": "QF3"
 !ENDQ = BCT -TCTR -NR
*    JLZ(ENDQ, 928)

     !JCT = JCT + 1
     !ENDJ = JCT - NSR

*    IF ( "JCT" GT "NEZ" ): !ENDJ = 1
*JLZ(ENDJ, 927)

*RETURN
*FLUX

$ ----------------------------------------------------------------
$
$  Section UPDT FLX2 - update flux estimate
$
$ ----------------------------------------------------------------

*(29 UPDT FLX2) FLX2

*XQTC AUS

DEFINE SKD = 1 FLUX DIST
DEFINE DQ = "TL" DQ
DEFINE SKU = "TL" SKU
DEFINE DQT = "TL" DQT
```

```
"TL" DQT = RTRAN( DQ )

!BCT = TCTR
*LABEL 555
    !BCT = BCT +1
    "TL" SKU = RPROD( DQT, SKD )

    TABLE,U(TYPE=-2): 1 SOUR K21 1
    TRANSFER(SOUR=SKU, L1="BCT", ILIM=1, JLIM="NEZ")

    !ENDT = BCT -TCTR -NR
*JLZ(ENDT, 555)
*RETURN
*FLX2
```

Appendix F

One-Dimensional Data Reduction Codes

The one-dimensional estimation codes were written in FORTRAN . Several of the programs use some common modules such as a forward conduction solution and material property functions which were compiled and linked separately.

Filename	Description	Section
`cf.f`	*Cook-Felderman*	F.1
`kd.f`	*Kendall-Dixon*	F.2
`dl.f`	*Diller*	F.3
`fd.f`	*Finite-Difference*	F.4
`rt.f`	*Rae-Taubee*	F.5
`si.f`	*Simple-Implicit*	F.6
`ir.f`	*Regularization*	F.7
`ie.f`	*Explicit-Regularization*	F.8
`cs.f`	*Constant-Specification* and *Exact-Matching*	F.9
`ls.f`	*Linear-Specification*	F.10

Object Modules and Subroutines (Section F.11)

Filename	Description
`impz.f`	implicit forward conduction solver for a vector of boundary fluxes and insulated interior
`pyrex.eng.f`	Pyrex properties in English units like slugs and btu's
`pyrex.si.f`	Pyrex properties in SI units
`macor.eng.f`	Macor properites in English units
`macor.si.f`	Macor properties in SI units
`upilex.eng.f`	Upilex properties in English units
`upilex.si.f`	Upilex properties in SI units

F.1 *Cook-Felderman*

```
program cf
Cfl -Fe\greg\bin\ cf.for

c  This program uses the Cook-Felderman technique to
c  estimate heat fluxes from a given temperature
c  history.

integer nmax
parameter (nmax=500)
real*8 dt, y(nmax), q0(nmax), den, buc1, q1, tm(nmax)
integer tct, ct, nt
real*8 pi, a, dum
real*8 rho, cp, k, betap

c  The file to be read contains the time value and
c  two temperature for that time.  Each temperature
c  stands for a different sensor location.

tct = 1
4 read(*,*,end=1) tm(tct),q0(tct),y(tct)
tct = tct + 1
goto 4
1 nt = tct - 1

c  The properties must be represented as constants
c  and are those for macor.  The units are
c  [c] = Btu / slug R
c  [k] = Btu / ft s R
c  [rho] = slug / ft^3

c rho = 4.91d0
c c = 4.41d0
c k = 2.42d-4

c rho = 1.73d0
c c = 0.7d0
c k = 0.97d0

c rho = 8991.0d0
c c = 385.0d0
c k = 385.6d0

c pyrex in english (see above) units
rho = 4.3211d0
c = 5.8886d0
k = 2.210702d-4

pi = 2.0d0*dasin(1.0d0)

c  calculate the constant from the C-F technique

a = 2.0d0*sqrt(rho*c*k/pi)
c a = 2.0d0*sqrt(rho(y(1))*cp(y(1))*k(y(1))/pi)

write(*,12) tm(1), q0(1), y(1), 0.0
do 2 tct = 2,nt
  buc1 = 0.0d0
  dt = (tm(tct)-tm(tct-1))
  do 3 ct = 2,tct
    if (ct.eq.tct) then
      den = sqrt(dt)
    else
      den = sqrt(tct*dt-ct*dt) +
     &             sqrt(tct*dt-(ct-1)*dt)
    endif
    buc1 = buc1 + (y(ct)-y(ct-1))/den
  3   continue
     q1 = a*buc1*(1.0d0+1.294d-3*(y(tct)-y(1)))
c   q1 = a*buc1
     write(*,12) tm(tct), q0(tct), y(tct), q1

  2 continue

  12 format (4(1x,e13.6))

   end
```

F.2 *Kendall-Dixon*

```
program kd
Cfl -Fe\greg\bin\ kd.for

c  This program uses the Kendall-Dixon technique to
c  estimate heat fluxes from a given temperature
c  history.
```

```
integer nmax
parameter (nmax=500)
real*8 dt, y(nmax), q0(nmax), den, buc1, q1(nmax), tm(nmax)
integer tct, ct, nt
real*8 pi, a
real*8 rho, cp, k, betap

c  The file to be read contains the time value and
c  two temperature for that time.  Each temperature
c  stands for a different sensor location.

tct = 1
4 read(*,*,end=1) tm(tct), q0(tct), y(tct)
tct = tct + 1
goto 4
1 nt = tct - 1

c  The properties must be represented as constants
c  and are those for macor.  The units are
c  [c] = Btu / slug R
c  [k] = Btu / ft s R
c  [rho] = slug / ft^3

c rho = 4.91d0
c c = 4.41d0
c k = 2.42d-4

c rho = 1.73d0
c c = 0.7d0
c k = 0.97d0

pi = 2.0d0*dasin(1.0d0)

c  calculate the constant from the K-D technique

a = sqrt(rho(y(1))*cp(y(1))*k(y(1))/pi)

write(*,12) tm(1), q0(1), y(1), 0.0
q1(1)=0.0d0
do 2 tct = 2,nt
  buc1 = 0.0d0
  dt = (tm(tct)-tm(tct-1))
  do 3 ct = 2,tct
    if (ct.eq.tct) then
      den = sqrt(dt)
    else
      den = sqrt(tct*dt-ct*dt) +
     &             sqrt(tct*dt-(ct-1)*dt)
    endif
    buc1 = buc1 + dt*(y(ct)+y(ct-1)-2.0d0*y(1))/den
3   continue
    q1(tct) = a*buc1
2 continue
 do tct=9,nt-9
    q2=-2.0d0*q1(tct-8)-q1(tct-4)+q1(tct+4)+2.0d0*q1(tct+8)
    q2=q2/40.0d0/dt
q2=q2*(1.0d0+betap(y(tct))*(y(tct)-y(1)))
    write(*,12) tm(tct), q0(tct), y(tct), q2
 enddo
11 format (1x,f12.8)
12 format (4(1x,e13.6))

 end
```

F.3 *Diller*

```
program dl

c  This program uses the Diller technique to
c  estimate heat fluxes from a given temperature
c  history.

integer nmax
parameter (nmax=500)
real*8 dt, y(nmax), q0(nmax), den, buc1, q1, tm(nmax)
integer tct, ct, nt
real*8 pi, a, dum
real*8 rho, cp, k, betap

c  The file to be read contains the time value and
c  two temperature for that time.  Each temperature
c  stands for a different sensor location.

tct = 1
4 read(*,*,end=1) tm(tct),q0(tct),y(tct)
```

```
tct = tct + 1
goto 4
1 nt = tct - 1

c  The properties must be represented as constants
c  and are those for macor.  The units are
c  [c] = Btu / slug R
c  [k] = Btu / ft s R
c  [rho] = slug / ft^3

c rho = 4.91d0
c c = 4.41d0
c k = 2.42d-4

c rho = 1.73d0
c c = 0.7d0
c k = 0.97d0

c rho = 8991.0d0
c c = 385.0d0
c k = 385.6d0

c pyrex in english (see above) units
rho = 4.3211d0
c = 5.8886d0
k = 2.210702d-4

pi = 2.0d0*dasin(1.0d0)

c  calculate the constant from the C-F technique

dt = tm(2)-tm(1)
a = sqrt(rho*c*k/pi/dt)
c a = 2.0d0*sqrt(rho(y(1))*cp(y(1))*k(y(1))/pi)

write(*,12) tm(1), q0(1), y(1), 0.0
do 2 tct = 2,nt-1
  buc1 = 0.0d0
  do 3 ct = 2,tct
     buc1 = buc1 + (y(ct)-y(ct-1))/sqrt(tct-ct+1.0)
  3   continue
c     q1 = a*buc1*(1.0d0+betap(y(tct))*(y(tct)-y(1)))
  q1 = a*(buc1+y(tct+1)-y(tct)+(y(tct)-y(tct-1))/3.0)
  q2 = q1*(1.0 + 1.294d-3*(y(tct)-y(1)))
     write(*,12) tm(tct), q0(tct), y(tct), q2

  2 continue

  12 format (4(1x,e13.6))

   end
```

F.4 *Finite-Difference*

```
program fd
Cfl -Fe\greg\bin\ fd.for
integer nx, nt
parameter(nx=200,nt=500)

real*8 t(nx), y(nt), dx(nx), dt, xt
real*8 tm(nt), q0(nt)
integer m(nx), iindx, gdepth, nlx, nq, ctx, mi(10)
real*8 lt(10), lk(10), lr(10), lc(10)
real*8 tdepth, q, xl

integer ct, nl, tpos

real*8 k, rtc


C-------------------------------------------------------------
C---- read in the information about layers



C**** read information for each layer

xt = 0.0d0
ct = 1
  32 read (*,*,end=31) tm(ct),q0(ct),y(ct)
ct = ct + 1
goto 32

  31 nq = ct-1


C-------------------------------------------------------------
C---- decide how to split up the fifty grid points into layers
```

```
C---- initial temperature

do 1 ct = 1,nx
  t(ct) = y(1)
  dx(ct) = 0.005d0/dfloat(nx)
  1 continue


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

write(*,90) tm(1),q0(1),y(1),0.0
do 4 tpos = 2,nq

  dt = tm(tpos) - tm(tpos-1)
  t(1) = y(tpos)

  call impldist(nx,t,dx,dt)

c   q = -k(t(1))*(-t(3)+4.0d0*t(2)-3.0d0*t(1))/(2.0d0*dx(1))
  est = rtc(t(1))*(y(tpos)-y(tpos-1))*dx(1)/(2.0d0*dt)
  q = est + k(t(1))*(t(1)-t(2))/dx(1)

  write(*,90) tm(tpos),q0(tpos),y(tpos),q
90   format(4(1x,e13.6))

  4 continue


  10 format(1x,f10.7,f10.7,f12.4,i3)
  11 format(a24,i2)
  12 format(a24,f10.9,1x,a12,1x,f15.9)
  13    format(1x,f16.4,3x,f16.10)
  14 format(a24,f12.7)
  15 format(a24,i4)
  16 format(f12.6)
  18 format(a24)
  19 format (1x,f7.3,3x,f10.4,3x,f10.3,3x,f10.3,3x,f10.3)

   end
```


F.5 *Rae-Taubee*

```
program rt

c  This program uses the Ray-Taulbee technique to
c  estimate heat fluxes from a given temperature
c  history. It employs a finite difference scheme
c  with a stretched coordinate system

integer ng, nq
parameter (ng=300,nq=500)

real*8 t(ng), to(ng), a(ng), b(ng), c(ng), u(ng)
real*8 tm(nq), temp(nq), q0(nq)
real*8 dt, f, dn, fo, t1, k, alpr, kr
real*8 resfo,res,tol,itm, dtemp, ki, tb
integer n, ctt, nt, pn, intt, ct
real*8 alp,th,rtbis,cp,rho
external th

pn = 2
intt = 10

ctt = 1
16 read(*,*,end=11) tm(ctt), q0(ctt), temp(ctt)
ctt = ctt + 1
goto 16
11 nt = ctt - 1

t1 = temp(1)
alpr = k(t1)/rho(t1)/cp(t1)
kr = k(t1)
ki = th(t1)

dn = 5.0d0/(ng-1)
f = 0.5d0
tol = 1.0d-7


a(1) = 0.0d0
b(1) = 1.0d0
c(1) = 0.0d0
a(ng) = 0.0d0
b(ng) = 1.0d0
c(ng) = 0.0d0
```

```
u(ng) = 0.0d0

do n = 1,ng
  to(n) = 0.0d0
enddo

write(*,81) tm(1), q0(1), temp(1), 0.0

do 2 ctt = 2,nt

  dt = (tm(ctt) - tm(ctt-1))/real(intt)
  dtemp = (temp(ctt)-temp(ctt-1))/real(intt)

  do 9 ct = 1,intt

    tb = temp(ctt-1) + ct*dtemp
    itm = tm(ctt-1) + ct*dt
    u(1) = 1.8d0*(th(tb)-ki)/kr

    resfo = 0.0d0

4     continue

    do 3 n = 2,ng-1

      if (tb.le.t1) then
        te = t1
      else
        te = rtbis(th,t1-1.0d0,tb*2.0d0,tol,ki,kr,t(n)/1.8d0)
      endif

      fo = alp(t(n)+t1)*dt/(alpr*dn*dn)
      a(n) = f*(fo - (n-1)*dt)
      b(n) = -2.0d0*f*fo - 4.0d0*itm
      c(n) = f*(fo + (n-1)*dt)
      u(n) = ((1.0d0-f)*2.0d0*fo - 4.0d0*itm)*to(n) -
     &               (1.0d0-f)*(fo - (n-1)*dt)*to(n-1) -
     &              (1.0d0-f)*(fo + (n-1)*dt)*to(n+1)
  3     continue

    call tridag(a,b,c,u,t,ng,ng)

    sumt = 0.0d0
    do n = 1,ng
      sumt = sumt + t(ng)
    enddo

    res = abs(resfo-fo)
    resfo = fo
    if (res.gt.tol) goto 4

    do n = 1,ng
      to(n) = t(n)
    enddo

9   continue

  q = -kr*(-3.0d0*t(1)+4.0d0*t(2)-t(3)) /
     &        (4.0d0*sqrt(alpr*tm(ctt))*dn)

  write(*,81) tm(ctt), q0(ctt), temp(ctt), q

  2 continue


81 format(4(1x,e13.6))

   end


real*8 function th(xn)
real*8 x, xn
x = xn/1.8d0
th = x*(x*(x*9.72433d-10-4.74326d-7)+2.43248d-4)
return
end

      FUNCTION rtbis(func,x1,x2,xacc,ki,kr,tn)
      INTEGER JMAX
      REAL*8 rtbis,x1,x2,xacc,func,ki,kr,tn
      EXTERNAL func
      PARAMETER (JMAX=40)
      INTEGER j
      REAL*8 dx,f,fmid,xmid
      fmid=(func(x2)-ki)/kr-tn
      f=(func(x1)-ki)/kr-tn
      if(f*fmid.ge.0.) pause 'root must be bracketed in rtbis'
      if(f.lt.0.)then
        rtbis=x1
        dx=x2-x1
```

```
      else
        rtbis=x2
        dx=x1-x2
      endif
      do 11 j=1,JMAX
        dx=dx*.5
        xmid=rtbis+dx
        fmid=(func(xmid)-ki)/kr-tn
        if(fmid.le.0.)rtbis=xmid
        if(abs(dx).lt.xacc .or. fmid.eq.0.) return
11    continue
      pause 'too many bisections in rtbis'
      END
C  (C) Copr. 1986-92 Numerical Recipes Software J!V%03#1y.

real*8 function alp(tn)
real*8 tn
real*8 k,rho,cp

alp = k(tn)/(rho(tn)*cp(tn))

return
end
```


F.6 *Simple-Implicit*

```
program si
Cfl -Fe\greg\bin\ si.for

integer nx, nt
parameter (nx=200,nt=500)

real*8 t(nx), to(nx), a(nx), b(nx), c(nx), u(nx)
real*8 tm(nt), temp(nt), q0(nt)
real*8 dt, dx, fo, t1, k, kr
real*8 rest,res,tol,itm, dtemp, ki, tb, te, sumt
integer n, ctt, nts, pn, intt, ct
real*8 alp, th, rtbis
external th

pn = 2

ctt = 1
16 read(*,*,end=11) tm(ctt), q0(ctt), temp(ctt)
ctt = ctt + 1
goto 16
11 nts = ctt - 1

dti = tm(2)-tm(1)
t1 = temp(1)
kr = k(t1)
ki = th(t1)
dx = 0.005d0/(nx-1)
intt = int(6.0d0*dti*alp(2.0d3)/dx/dx)

tol = 1.0d-8

a(1) = 0.0d0
b(1) = 1.0d0
c(1) = 0.0d0
a(nx) = 0.0d0
b(nx) = 1.0d0
c(nx) = 0.0d0
u(nx) = 0.0d0

do n = 1,nx
  to(n) = 0.0d0
enddo

write(*,81) tm(1), q0(1), temp(1), 0.0
do 2 ctt = 2,nts

  dt = (tm(ctt) - tm(ctt-1))/real(intt)
  dtemp = (temp(ctt)-temp(ctt-1))/real(intt)

  do 9 ct = 1,intt

    tb = temp(ctt-1) + ct*dtemp
    itm = tm(ctt-1) + ct*dt
    u(1) = 1.8d0*(th(tb)-ki)/kr

    rest = 0.0d0

4     continue
```

```
    do 3 n = 2,nx-1

      if (tb.le.t1) then
        te = t1
      else
        te = rtbis(th,t1-1.0d0,tb*2.0d0,tol,ki,kr,t(n)/1.8d0)
      endif

      fo = alp(te)*dt/(dx*dx)
      a(n) = -fo
      b(n) = 2.0d0*fo + 1.0d0
      c(n) = -fo
      u(n) = to(n)
  3     continue

    call tridag(a,b,c,u,t,nx,nx)

    sumt = 0.0d0
    do n = 1,nx
      sumt = sumt + t(nx)
    enddo
    res = abs(rest-sumt)
    rest = sumt
    if (res.gt.tol) goto 4

    do n = 1,nx
      to(n) = t(n)
    enddo

9   continue

  q = -kr*(-3.0d0*t(1)+4.0d0*t(2)-t(3)) / (dx+dx)

  write(*,81) tm(ctt), q0(ctt), temp(ctt), q

  2 continue


81 format(4(1x,e13.6))

   end


real*8 function th(xn)
real*8 x, xn
x = xn/1.8d0
th = x*(x*(x*9.72433d-10-4.74326d-7)+2.43248d-4)
return
end

      FUNCTION rtbis(func,x1,x2,xacc,ki,kr,tn)
      INTEGER JMAX
      REAL*8 rtbis,x1,x2,xacc,func,ki,kr,tn
      EXTERNAL func
      PARAMETER (JMAX=40)
      INTEGER j
      REAL*8 dx,f,fmid,xmid
      fmid=(func(x2)-ki)/kr-tn
      f=(func(x1)-ki)/kr-tn
      if(f*fmid.ge.0.) pause 'root must be bracketed in rtbis'
      if(f.lt.0.)then
        rtbis=x1
        dx=x2-x1
      else
        rtbis=x2
        dx=x1-x2
      endif
      do 11 j=1,JMAX
        dx=dx*.5
        xmid=rtbis+dx
        fmid=(func(xmid)-ki)/kr-tn
        if(fmid.le.0.)rtbis=xmid
        if(abs(dx).lt.xacc .or. fmid.eq.0.) return
11    continue
      pause 'too many bisections in rtbis'
      END
C  (C) Copr. 1986-92 Numerical Recipes Software J!V%03#1y.

real*8 function alp(tn)
real*8 tn
real*8 k,rho,cp

alp = k(tn)/(rho(tn)*cp(tn))

return
end
```

F.7 *Regularization*

```
program ir

integer nx, nt, nr
parameter(nx=100,nt=500,nr=3)

real*8 t(nx), y(nt), dx(nx), dt, xt, v(nr)
real*8 t1(nx), ty(nr), ts(nr), x(nr,nr), tm(nt), h(nr,nr)
real*8 c(nr), a(nr,nr), p(nt), q0(nt)
integer nq
real*8 qsave
real*8 qi(0:nr), sens, tol, one, alpha
real*8 h0(nr,nr), h1(nr,nr), h2(nr,nr), w0, w1, w2
real*8 d, err1, err2, w, relax, pent
integer indx(nr)

integer ct, tpos, dqct, i, j

c read(*,*) alpha
alpha = 1.0d-4

ct = 1
  32 read (*,*,end=31) tm(ct), q0(ct), y(ct)
ct = ct + 1
goto 32

  31 nq = ct-1

tol = 1.0d-4
one = 1.0d0
sens = 0.00001d0
xt = 0.0d0
w0 = 0.0d0
w1 = 1.0d0
w2 = 0.0d0
relax = 1.0d0
pent = 2.0d0 * sqrt ( 1.17438d-4 * tm(nq) )

do ct = 1,nx
   dx(ct) = pent/dfloat(nx)
enddo

do ct = 1,nx
  t(ct) = y(1)
enddo

do i = 0,nr
  qi(i) = 0.0d0
enddo

do ct = 1,nt
   p(ct) = one
enddo
p(100) = 1.0d-2

C-------------------------------------------------------
C---- inverse parameters

        INCLUDE 'regular2.f'

do i = 1,nr
  qi(i) = 1.0d-3
enddo

C-------------------------------------------------------
C---- begin inverse loop section

write(*,11) tm(1),q0(1),y(1),0.0

do 4 tpos = 2,nq-nr+1

  dt = tm(tpos) - tm(tpos-1)
  dqct = 0
  w = relax

C---- iteration loop

73   continue
  dqct = dqct + 1

     do ct = 1,nx
       t1(ct) = t(ct)
     enddo

  call imp1(nx,nr,t1,qi,dx,dt,ts,xt)

  do j = 1,nr
       do ct = 1,nx
         t1(ct) = t(ct)
```

```
       enddo
       qsave = qi(j)
       qi(j) = qsave*(one+sens)+sens

    call imp1(nx,nr,t1,qi,dx,dt,ty,xt)

    do i = 1,nr
      x(i,j) = (ty(i) - ts(i))/(qi(j)-qsave)
    enddo

           qi(j) = qsave
  enddo

C---- calculate lhs (A matrix)

  do i = 1,nr
    do j = 1,nr
      a(i,j) = 0.0d0
      do ct = 1,nr
        a(i,j) = a(i,j) + x(ct,i)*p(tpos+ct-1)*x(ct,j)
      enddo
      a(i,j) = a(i,j) + h(i,j)
    enddo
  enddo

C---- calculate temperature difference

  do i = 1,nr
    v(i) = y(tpos+i-1) - ts(i)
  enddo

C---- calculate rhs (c vector)

  do i = 1,nr
    c(i) = 0.0d0
    do ct = 1,nr
      c(i) = c(i) + x(ct,i)*p(tpos+ct-1)*v(ct) - h(i,ct)*qi(ct)
    enddo
  enddo

C---- solve the matrix equations (A dq = c)

  call ludcmp(a,nr,nr,indx,d)
  call lubksb(a,nr,nr,indx,c)

C---- calculate err (dq)

  err1 = 0.0d0
  err2 = 0.0d0
  do i = 1,nr
    err1 = err1 + abs(c(i)/qi(i))
    err2 = err2 + abs(c(i))
  enddo

C---- update count and flux "guess"
  if (dqct.gt.5) w = relax/10.0d0
  if (dqct.gt.15) w = relax/30.0d0

  do i = 1,nr
     qi(i) = qi(i) + w*c(i)
  enddo

  if (dqct.lt.40) then
     if ((err1.gt.tol).and.(err2.gt.tol)) goto 73
  endif

C---- direct solution for correct flux and write out results

  call imp2(nx,t,qi(0),qi(1),dx,dt)
  write(*,11) tm(tpos),q0(tpos),y(tpos),qi(1)

C---- bump up flux "guess" for next time step

          do i = 0,nr-1
            qi(i) = qi(i+1)
          enddo

  4 continue

  10 format(1x,f10.7,f10.7,f12.4,i3)
  11 format(4(3x,e13.6))
  13    format(1x,f16.4,3x,f16.10)
  14 format(a24,f12.7)
  15 format(a24,i4)
  16 format(f12.6)
  18 format(a24)
  19 format (1x,6(1x,e12.6))

   end
```

```
C include impliter subroutine and property functions

       INCLUDE 'matslv.f'
c       INCLUDE 'impz.f'
```

F.8 *Explicit-Regularization*

```
      program ie

      integer nx, nt
      parameter (nx=100, nt=2050)

      real*8 tm(nt), y(nt), q0(nt)
      real*8 t(nx) , dx(nx), to(nx), p
      real*8 dt, sens, q1, q2, lx, x, x2, x1, q2s, dq, a, y1
      integer nq, cnx, cnt, cni

      real*8 k, rho, cp, alpha

c      read(*,*) a
      a = 1.0d-4

      ct = 1
 2    read(*,*,end=1) tm(ct), q0(ct), y(ct)
      ct = ct + 1
      goto 2

 1    nq = ct - 1

      y1 = y(1)
      do cnx = 1, nx
         t(cnx) = y1
      enddo

      lx = 2.0d0*sqrt(k(y1)/rho(y1)/cp(y1)*tm(nq))
      do cnx = 1, nx
         dx(cnx) = lx / real(nx)
      enddo

      sens = 0.001d0

      q1 = q0(1)
      q2 = 1.0d0

      write(*,9) tm(1), q0(1), y(1), q1

      do cnt = 2, nq

         p = 1.0d0
c         if (cnt.eq.100) p = 1.0d-4

         do cnx = 1, nx
            to(cnx) = t(cnx)
         enddo

         dt = tm(cnt) - tm(cnt-1)
         cni = 0

 4       cni = cni + 1

         do cnx = 1, nx
            t(cnx) = to(cnx)
         enddo

         call imp2(nx,t,q1,q2,dx,dt)
         x1 = t(1)

         if (abs(y(cnt)-x1).lt.0.1) goto 5
         if (cni.gt.40) goto 5

         do cnx = 1, nx
            t(cnx) = to(cnx)
         enddo
         q2s = q2 * (1.0d0 + sens)

         call imp2(nx,t,q1,q2s,dx,dt)
         x2 = t(1)

         x = (x2 - x1) / (q2s-q2)
         dq = (x*p*(y(cnt)-x1)+a*(q1-q2))/(a+x*x*p)
         q2 = q2 + dq

         goto 4

 5       write(*,9) tm(cnt), q0(cnt), y(cnt), q2
         q1 = q2
```

```
      enddo

 9    format (4(3x,e13.6))

      end
```

F.9 *Constant-Specification*

```
c program: cs - constant specification
c
c author: Greg Walker
c
c purpose: estimate heat fluxes from surface temperature measurements
c          using the function specification method with a constant
c          function

program cs

c define array sizes
c nx - number of nodes
c nt - maximum number of time steps
c nr - number of future time steps

integer nx, nt, nr
parameter(nx=100,nt=500,nr=3)

c define distribution variables
c t - calculated temperature distribution
c dx - node spacing
c t1 - temporary calculated distribution (so t doesn't get destroyed)
c xt - sensor location (included for historical reasons)

real*8 t(nx), dx(nx), t1(nx), xt

c define history variables
c y - temperature history on the surface
c ty - calculated temperature measurements
c ts - calculated temperature measurements
c tm - time
c dt - time step
c x - sensitivity
c q0 - known flux

real*8 y(nt), ty(nr), ts(nr), tm(nt), dt, x(nr), q0(nt)

c define other variables
c nq - number of times read
c qsave - temporary holding value for qi
c qi - current flux estimate
c sens - sensitivity percentage
c tol - convergent criterion
c one - one, unity, uno, solo, solitary, numerically alone
c dq - flux correction
c dq1 - numerator of flux correction
c dq2 - denominator of flux correction

integer nq
real*8 qsave
real*8 qi(0:nr),sens,tol,one
real*8 dq, dq1, dq2

c define counters
c ct - dummy
c i - dummy
c tpos - counts the measurements
c dqct - counts iteration loops

integer ct, tpos, i, dqct

c read data

ct = 1
  32 read (*,*,end=31) tm(ct),q0(ct),y(ct)
ct = ct + 1
goto 32

  31 nq = ct-1

c initialize variables

tol = 1.0d-3
one = 1.0d0
sens = 0.00001d0

do ct = 1,nx
   dx(ct) = 0.003d0/dfloat(nx)
enddo
```

```
do ct = 1,nx
  t(ct) = y(1)
enddo

do i = 0,nr
  qi(i) = 0.0d0
enddo

C-------------------------------------------------------
C---- begin inverse loop section

write(*,11) tm(1),q0(1),y(1),0.0

do 4 tpos = 2,nq-nr+1

  dt = tm(tpos) - tm(tpos-1)

C---- direct solution and temp changes

  dqct = 0
73   continue
  dqct = dqct + 1

     do ct = 1,nx
       t1(ct) = t(ct)
     enddo

  call imp1(nx,nr,t1,qi,dx,dt,ts,xt)

  do ct = 1,nx
     t1(ct) = t(ct)
  enddo

  qsave = qi(1)
  do i = 0,nr
     qi(i) = qsave*(one+sens) + sens
  enddo

  call imp1(nx,nr,t1,qi,dx,dt,ty,xt)

  do i = 1,nr
     x(i) = (ty(i) - ts(i))/(qi(i)-qsave)
  enddo

do i = 0,nr
   qi(i) = qsave
enddo

dq1 = 0.0d0
dq2 = 0.0d0

do i = 1,nr
   dq1 = dq1 + (y(tpos+i-1)-ts(i))*x(i)
   dq2 = dq2 + x(i)*x(i)
enddo

dq = dq1/dq2

do i = 0,nr
   qi(i) = qi(i) + dq
enddo

if (dqct.lt.20) then
   if (abs(dq).gt.tol) goto 73
endif

call imp2(nx,t,qi(0),qi(1),dx,dt)

write(*,11) tm(tpos),q0(tpos),y(tpos),qi(1)

4 continue

 close(2)


10 format(1x,f10.7,f10.7,f12.4,i3)
11 format(4(1x,e13.6))
13    format(1x,f16.4,3x,f16.10)
14 format(a24,f12.7)
15 format(a24,i4)
16 format(f12.6)
18 format(a24)
19 format (1x,6(1x,e12.6))

 end
```

F.10 *Linear-Specification*

```
c program: ls - function specification
c
c author: Greg Walker
c
c purpose: estimate heat fluxes from surface temperature measurements
c          using the function specification method with a linear
c          function

program ls

c define array sizes
c nx - number of nodes
c nt - maximum number of time steps
c nr - number of future time steps

integer nx, nt, nr
parameter(nx=100,nt=500,nr=3)

c define distribution variables
c t - calculated temperature distribution
c dx - node spacing
c t1 - temporary calculated distribution (so t doesn't get destroyed)
c xt - sensor location (included for historical reasons)

real*8 t(nx), dx(nx), t1(nx), xt

c define history variables
c y - temperature history on the surface
c ty - calculated temperature measurements
c ts - calculated temperature measurements
c tm - time
c dt - time step
c x - sensitivity
c q0 - known flux

real*8 y(nt), tm(nt), dt, q0(nt)
real*8 xs, ty(nr), ts(nr), x(nr)

c define other variables
c nq - number of times read
c qsave - temporary holding value for qi
c qi - current flux estimate
c sens - sensitivity percentage
c tol - convergent criterion
c one - one, unity, uno, solo, solitary, numerically alone
c dq - flux correction
c dq1 - numerator of flux correction
c dq2 - denominator of flux correction

integer nq
real*8 qs(0:nr)
real*8 qi(0:nr),sens,tol,one
real*8 dq, dq1, dq2

c define counters
c ct - counts to nx
c i - counts to nr
c tpos - counts the measurements
c dqct - counts iterative loops

integer ct, tpos, i, dqct

c read data

ct = 1
  32 read (*,*,end=31) tm(ct),q0(ct),y(ct)
ct = ct + 1
goto 32

  31 nq = ct-1

c initialize variables

tol = 1.0d-3
one = 1.0d0
sens = 0.00001d0

do ct = 1,nx
   dx(ct) = 0.003d0/dfloat(nx)
enddo

do ct = 1,nx
  t(ct) = y(1)
enddo

do i = 0,nr
  qi(i) = 0.0d0
enddo
```

```
C-------------------------------------------------------
C---- begin inverse loop section

write(*,11) tm(1),q0(1),y(1),0.0

do 4 tpos = 2,nq-nr+1

  dt = tm(tpos) - tm(tpos-1)

c iteration loop
  dqct = 0
73   continue
  dqct = dqct + 1

     do ct = 1,nx
       t1(ct) = t(ct)
     enddo

  call imp1(nx,nr,t1,qi,dx,dt,ts,xt)

  do ct = 1,nx
     t1(ct) = t(ct)
  enddo

  qs(1) = qi(1)
  qi(1) = qs(1)*(one+sens)+sens

  do i = 2,nr
     qs(i) = qi(i)
     qi(i) = 2.0d0*qi(i-1) - qi(i-2)
  enddo

  call imp1(nx,nr,t1,qi,dx,dt,ty,xt)

  do i = 1,nr
     x(i) = (ty(i) - ts(i))/(qi(i)-qs(i))
  enddo

  do i = 1,nr
     qi(i) = qs(i)
  enddo

  dq1 = 0.0d0
  dq2 = 0.0d0

xs = 0.0d0
do i = 1,nr
   xs = xs + x(i)
   dq1 = dq1 + (y(tpos+i-1)-ts(i))*xs
   dq2 = dq2 + xs*xs
enddo

dq = dq1/dq2

qi(1) = qi(1) + dq
do i = 2,nr
   qi(i) = 2.0d0*qi(i-1) - qi(i-2)
enddo

if (dqct.lt.20) then
   if (abs(dq).gt.tol) goto 73
        endif

call imp2(nx,t,qi(0),qi(1),dx,dt)

write(*,11) tm(tpos),q0(tpos),y(tpos),qi(1)

do i = 0,nr-1
   qi(i) = qi(i+1)
enddo
qi(nr) = 2.0*qi(nr-1)-qi(nr-2)

4 continue

 close(2)


10 format(1x,f10.7,f10.7,f12.4,i3)
11 format(4(1x,e13.6))
13    format(1x,f16.4,3x,f16.10)
14 format(a24,f12.7)
15 format(a24,i4)
16 format(f12.6)
18 format(a24)
19 format (1x,6(1x,e12.6))

 end
```

F.11 Object Modules and Subroutines

```
C*********************************************************
      subroutine imp1(nx,ni,t,qv,dx,dti,ts,xt)
C*********************************************************
C   Implicit routine to determine temperature history
C   Thomas Algorithm
C   Semi-Infinite Slab
C   Specified Flux at surface
C   Insulated or constant Temperature Interior
C   Variable Grid Spacing
C   Temperature variant thermal Properties
C   Layer Interface must coincide with node

C define passed varibles
      integer nx, ni
      real*8 t(nx), dx(nx), dti, qv(0:ni), ts(ni), xt

      integer nmax
      real*8 tol
      parameter (nmax=200,tol=1.0d-4)

      integer ct, ctr, ctil, tl, n
      real*8 ix, xc, q
      real*8 t1, dt
      real*8 to(nmax), a(nmax), b(nmax), c(nmax), r(nmax)

C variables for the matrix building section
      real*8 dxr,dxl,dxp,tw,te
      real*8 rc,rcr,rcl,kr,kl,for,fol


C define functions
      real*8 rho,cp,k,rtc

      tl = int(8.0d0*dti*k(t(1))/dx(1)/dx(1)/rho(t(1))/cp(t(1)))
      dt = dti / dfloat(tl)

      ix = 0
      xc = 0.0d0
81    ix = ix + 1
      xc = xc + dx(ix)
      if (xc.lt.xt) goto 81

      do 10 ctr = 1,ni

do 22 ctil = 1,tl

  n = 0
         q = qv(ctr-1) + ctil*(qv(ctr)-qv(ctr-1))/dfloat(tl)

C save old temperature values
do ct = 1,nx
  to(ct) = t(ct)
enddo

4 t1 = t(1)

C---- this section builds the matrix to be solved
        INCLUDE 'implicit.f'

call tridag(a,b,c,r,t,nx,nx)

if (n.eq.200) stop 'temp dist not converging - imp1'
n = n + 1

if (abs(t1-t(1))/t(1).gt.tol) goto 4

22 continue

ts(ctr) = t(ix+1) - (xc-xt)*(t(ix+1)-t(ix))/dx(ix)

10    continue

      return
      end

C*********************************************************
      subroutine imp2(nx,t,q1,q2,dx,dti)
C*********************************************************
C   Implicit routine to determine temperature distribution
C   Thomas Algorithm
C   Semi-Infinite Slab
C   Specified Flux at surface
C   Insulated or constant Temperature Interior
C   Variable Grid Spacing
C   Temperature variant thermal Properties
C   Layer Interface must coincide with node

C define passed varibles
```

```
      integer nx
      real*8 t(nx), dx(nx), dti, q1, q2

      integer nmax
      real*8 tol
      parameter (nmax=400,tol=1.0d-4)

      integer ct, ctil, tl, n
      real*8 t1, dt
      real*8 to(nmax), a(nmax), b(nmax), c(nmax), r(nmax)

C variables for the matrix building section
      real*8 dxr,dxl,dxp,tw,te
      real*8 rc,rcr,rcl,kr,kl,for,fol


C define functions
      real*8 rho,cp,k,rtc

      tl = int(8.0d0*dti*k(t(1))/dx(1)/dx(1)/rho(t(1))/cp(t(1)))
      dt = dti / dfloat(tl)

      do 22 ctil = 1,tl

n=0
        q =  q1 + ctil*(q2-q1)/dfloat(tl)
        q = q1

C save old temperature values
do ct = 1,nx
  to(ct) = t(ct)
enddo

4 t1 = t(1)

C---- this section builds the matrix to be solved
        INCLUDE 'implicit.f'

call tridag(a,b,c,r,t,nx,nx)

if (n.eq.200) stop 'temp dist not converging - imp2'
n = n + 1

if (abs(t1-t(1))/t(1).gt.tol) goto 4

22    continue

      return
      end



C*********************************************************
        subroutine impldist(nx,t,dx,dti)
C*********************************************************
C         Implicit routine to determine temperature distribution
C         Thomas Algorithm
C         Semi-Infinite Slab
C         Specified Temperature at surface
C         Insulated or constant Temperature Interior
C         Temperature dependent thermal Properties

Cdefine passed varibles
        integer nx
        real*8 t(nx), dx(nx), dti

        integer nmax
        real*8 tol
        parameter (nmax=501,tol=1.0d-9)

        integer ct, tstep, tloop
        real*8 dxp, rcr, rcl, for, fol, kr, kl, rc
        real*8 te, tw, dxl, dxr, t1, dt, delt
        real*8 to(nmax), a(nmax), b(nmax), c(nmax), r(nmax)

C define functions
        real*8 rho,cp,k,rtc

delt = 100.0d0
        tloop = int(abs(delt)) + 30
        dt = dti / dfloat(tloop)

        do 8 tstep = 1,tloop

C save old temperature values
5       do 2 ct = 1,nx
2         to(ct) = t(ct)

C-----------------------------------------------------------
C start with node 1 (temp boundary)
```

```
  4     t1 = t(1)

        a(1) = 0.0d0
        b(1) = 1.0d0
        c(1) = 0.0d0
        r(1) = t1

C interior boundary (specified temperature)
       a(nx) = 0.0d0
       b(nx) = 1.0d0
       c(nx) = 0.0d0
       r(nx) = to(nx)

C interior boundary (insulated)
c        a(nx) = -1.0d0
c        b(nx) = 1.0d0
c        c(nx) = 0.0d0
c        r(nx) = 0.0d0

        do 1 ct = 2,nx-1

          dxr = dx(ct)
          dxl = dx(ct-1)
          dxp = (dxr+dxl)/2.0d0

  rcr = rtc(t(ct))
  rcl = rtc(t(ct))
          rc = (rcl*dxl + rcr*dxr)/(dxr+dxl)

          te = (t(ct)+t(ct+1))/2.0d0
          kr = k(te)
          tw = (t(ct)+t(ct-1))/2.0d0
          kl = k(tw)

          for = kr*dt/(rc*dxr*dxp)
          fol = kl*dt/(rc*dxl*dxp)

          a(ct) = -fol
          b(ct) = 1 + fol + for
          c(ct) = -for
          r(ct) = to(ct)

  1     continue

        call tridag(a,b,c,r,t,nx,nx)

        if (abs(t1-t(1)).gt.tol) goto 4

8 continue

        return
        end

SUBROUTINE tridag(a,b,c,r,u,n,nx)
INTEGER n,NMAX
REAL*8 a(n),b(n),c(n),r(n),u(n)
PARAMETER (NMAX=400)
C   Solves
INTEGER j
REAL*8 bet,gam(NMAX)
if(b(1).eq.0.0d0)pause 'tridag: rewrite equations'
bet=b(1)
u(1)=r(1)/bet
do 11 j = 2,nx
  gam(j) = c(j-1)/bet
  bet = b(j)-a(j)*gam(j)
  if(bet.eq.0.0d0)pause 'tridag failed'
  u(j) = (r(j)-a(j)*u(j-1))/bet
  11 continue
do 12 j = nx-1,1,-1
  u(j) = u(j)-gam(j+1)*u(j+1)
  12 continue
return
END
```

```
C*********************************************************
real*8 function k(t)
real*8 t

c conductivity of pyrex
```

```
c input temperature in Rankine
c return conductivity in Btu/ft^2 ???


k = 2.43248272d-4 + t*(-5.2702971d-7 + t*9.0040103d-10)

   return
end

C***********************************************************
real*8 function rtc(t)
real*8 t

rtc = -2.83815917d0 + t*(7.9081613d-2 + t*(-5.8951355d-5
     .        + t*1.7891521d-8))

return
end

C***********************************************************
real*8 function cp(t)
real*8 t

cp = -6.5681505d-1 + t*(1.8301297d-2 + t*(-1.36426941d-5
     .       + t*4.093675d-9))

return
end

C***********************************************************
real*8 function rho(t)
real*8 t

rho = 4.3210934d0

return
end

C*********************************************************
real*8 function betap(t)
real*8 t

c beta' correction coefficient of pyrex
c input temperature in Rankine (really dont need)
c return beta' (1/R)

betap = 1.294d-3
c betap = 4.194d-3
c betap = 0.0d0

   return
end
```

```
C*********************************************************
real*8 function k(t)
real*8 t

c conductivity of macor
c input temperature in K
c return conductivity in W/mK


k = 0.33889d0 + t*(7.4682d-4 + t*(-1.6118d-5 + t*1.2376d-7))

   return
end

C***********************************************************
real*8 function rtc(t)
real*8 t

rtc = 2.901d7 + t*(6409.46d0 + t*(-3.85036d0 + t*(-1.21088d-12)))


return
end

C***********************************************************
real*8 function cp(t)
real*8 t

cp = 114.04d0 + t*(2.5196d0 + t*(-1.5136d-3))

return
end

C***********************************************************
real*8 function rho(t)
real*8 t
```

```
rho = 2543.84d0 + t*(-8.0d-12)

return
end

C*********************************************************
real*8 function betap(t)
real*8 t

c beta' correction coefficient of pyrex
c input temperature in Rankine (really dont need)
c return beta' (dimensionless


betap = 754.2d0 + t*(3.7201d0 + t*(-2.4883d-3))

   return
end

C*********************************************************
real*8 function alpha(t)
real*8 t

alpha = 1.3003d-6 + t*(-2.2523d-9 + t*1.8571d-12)


   return
end
```

```
C*********************************************************
real*8 function k(t)
real*8 t

c conductivity of upilex
c input temperature in K
c return conductivity in W/mK


k = -0.26918 + t*(3.6348d-3 + t*(-7.2432d-6 + t*5.0056d-9))
```

```
   return
end

C*********************************************************
real*8 function rho(t)
real*8 t

rho = 1490.0d0

return
end

C*********************************************************
real*8 function cp(t)
real*8 t

cp = -2258.2 + t*(19.492 + t*(-3.7267d-2 + t*2.7812d-5))

return
end

C*********************************************************
real*8 function rtc(t)
real*8 t

rtc = rho(t) * cp(t)

return
end



C*********************************************************
real*8 function betap(t)
real*8 t

c beta' correction coefficient of pyrex
c input temperature in Rankine (really dont need)
c return beta' (dimensionless


betap = -1068.7 + t*(10.619 + t*(-2.0458d-2 + t*(1.4544d-5)))

   return
end
```

```
C*******************************************************
real*8 function alpha(t)
real*8 t

alpha = k(t)/rho(t)/cp(t)

   return
end
```

Vita

The author, Greg Walker, began his academic career at Auburn University in 1985. A short five and a half years later he could be called a college graduate, having received his Bachelors of Science in Mechanical Engineering. His education continued at Auburn under the guidance of Dr. Daniel Mackowski in his quest for higher degrees. Upon successful completion of his Master's of Science degree in 1993, the lure of an academic career (and fear of the "real world") caused him to seek out other schooling opportunities. Upon arrival at Virginia Tech to pursue his Ph.D. under Dr. Elaine Scott, he was awarded a Fellowship from NASA Langley Research Center. This document now signifies the successful completion of his terminal degree in December, 1997.

Permanent Address: 2002 Broken Oak Drive
Blacksburg, Virginia 24060

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Greg Walker, Department of Mechanical Engineering, Virginia Tech.