PNL-6474
UC-70

4F

Nuclear Waste Treatment Program

# Application of the TEMPEST Computer Code to Canister-Filling Heat Transfer Problems

R. K. Farnsworth
D. W. Faletti
M. J. Budden

March 1988

Prepared for the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Operated for the U.S. Department of Energy
by Battelle Memorial Institute

Battelle

PNL-6474

## DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Battelle Memorial Institute, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government of any agency thereof, or Battelle Memorial Institute. The views and opinions of authors expressed herein do not necessarly state or reflect those of the United States Government or any agency thereof, or Battelle Memorial Institute.

PACIFIC NORTHWEST LABORATORY
*operated by*
BATTELLE MEMORIAL INSTITUTE
*for the*
UNITED STATES DEPARTMENT OF ENERGY
*under Contract DE-AC06-76RLO 1830*

Printed in the United States of America
Available from
National Technical Information Service
United States Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22161

NTIS Price Codes
Microfiche A01

Printed Copy

| Pages | Price Codes |
|---|---|
| 001-025 | A02 |
| 026-050 | A03 |
| 051-075 | A04 |
| 076-100 | A05 |
| 101-125 | A06 |
| 126-150 | A07 |
| 151-175 | A08 |
| 176-200 | A09 |
| 201-225 | A010 |
| 226-250 | A011 |
| 251-275 | A012 |
| 276-300 | A013 |

PNL-6474
UC-70

Nuclear Waste Treatment Program

# APPLICATION OF THE TEMPEST COMPUTER CODE TO CANISTER-FILLING HEAT TRANSFER PROBLEMS

R. K. Farnsworth
D. W. Faletti
M. J. Budden

March 1988

Prepared for
the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Richland, Washington 99352

## SUMMARY

Pacific Northwest Laboratory (PNL) researchers used the TEMPEST computer code to simulate thermal cooldown behavior of nuclear waste glass after it was poured into steel canisters for long-term storage. The objective of this work was to determine the accuracy and applicability of the TEMPEST code when used to compute canister thermal histories.

First, experimental data were obtained to provide the basis for comparing TEMPEST-generated predictions. Five canisters were instrumented with appropriately located radial and axial thermocouples. The canisters varied in diameter (12 in., 13 in., and 24 in.), height (45 in., 56 in., and 85 in.), internal insulation thickness (0 to 1/2 in.), and external insulation thickness (0 to 3 in.). The canisters were filled using the pilot-scale ceramic melter (PSCM) at PNL. Each canister was filled in either a continuous or a batch filling mode. One of the canisters was also filled within a turntable simulant (a group of cylindrical shells with heat transfer resistances similar to those in an actual melter turntable). This was necessary to provide a basis for assessing the ability of the TEMPEST code to also model the transient cooling of canisters in a melter turntable.

After the data were obtained from the PSCM runs, two versions of TEMPEST--the batch fill (L4X) and the continuous fill (M)--were used to simulate the experimental conditions. The thermal histories predicted by these TEMPEST versions were then compared to the experimentally-measured data.

Researchers found that the batch-fill model (L4X) predicted canister centerline temperatures that agreed closely with experimental temperatures, varying from only 50 to 125°C over most of the canister cooldown period. However, during periods immediately following a glass pour, temperatures were overpredicted by as much as 250°C.

The continuous-fill model, Version M, was found to predict temperatures with more accuracy. Variation from experimentally-measured temperatures was within 50°C or better over most of the cooldown. Close agreement was also achieved for temperatures at other radial positions away from the centerline and for times long after the glass surface rose above a given elevation.

The turntable simulant experiment demonstrated that TEMPEST can adequately model the asymmetric temperature field caused by the turntable geometry. Further, TEMPEST can acceptably predict the canister cooling history within a turntable, despite code limitations in computing simultaneous radiation and convection heat transfer between shells, along with uncertainty in stainless-steel surface emissivities.

Based on the successful performance of TEMPEST Version M, development was initiated to incorporate 1) full viscous glass convection, 2) a dynamically adaptive grid that automatically follows the glass/air interface throughout the transient, and 3) a full enclosure radiation model to allow radiation heat transfer to non-nearest neighbor cells. At the end of FY 1987, the first feature was fully incorporated and the second feature had been partially accomplished. This advanced version of TEMPEST, designated Version N29, is still under development.

Based on the results obtained using TEMPEST versions L4X and M, recommendations for future work include continuing the developmental efforts for Version N29 and improving the accuracy of materials properties and initial conditions inherent in the glass-filling process.

## CONTENTS

## FIGURES

## TABLES

# 1.0 INTRODUCTION

The liquid-fed ceramic melter (LFCM) is being developed in the United States and abroad as a promising method for treatment of high-level radioactive wastes. In the LFCM process, radioactive materials are combined with other glass-forming materials. The resultant mixture is then melted to form a chemically stable glass. This molten glass will subsequently be poured into canisters; the canisters will then be placed in a waste repository.

One of the factors that has been considered in determining the suitability of glass as a disposal material is the extent of glass devitrification and cracking in the waste canisters. These behaviors are determined largely by the cooling history of the glass. Hence, a method is needed to accurately compute the thermal history of canisters filled in the LFCM, from the time that canister filling begins until the canisters reach thermal equilibrium with the environment.

To investigate the code's effectiveness in predicting canister thermal behavior, Pacific Northwest Laboratory (PNL)[a] researchers applied the TEMPEST computer code to various canister-filling scenarios. This study was conducted for the U.S. Department of Energy (DOE) through the Nuclear Waste Treatment Program at PNL.

The TEMPEST computer code (Trent, Eyler, and Budden 1983[b]; Trent and Eyler 1985; Eyler and Trent 1984), solves the partial differential equations governing the conservation of mass, momentum, and thermal energy in either cylindrical or cartesian geometries. The code is well-documented and has undergone an extensive verification and benchmarking effort to establish its accuracy and range of validity. Despite its relatively large size (>30,000 lines), TEMPEST is flexible and relatively easy to use.

[a] Operated for the U.S. Department of Energy by Battelle Memorial Institute under Contract DE-AC06-76RLO 1830.

[b] Discussion is contained in Trent, D.S., L. L. Eyler, and M. J. Budden. 1983. TEMPEST - A Three-Dimensional Time-Dependent Computer Program for Hydrothermal Analysis. Volume I: Numerical Methods and Input Instruction. Pacific Northwest Laboratory, Richland, Washington.

This report documents the comparison of TEMPEST-generated thermal history predictions with experimentally derived data for several different canister designs and filling scenarios. The report also provides a description of PNL's work to date on modifications to TEMPEST to improve the code's predictive capabilities when used to model canister filling.

The conclusions and recommendations stemming from this study are presented in Chapter 2. In Chapter 3, the TEMPEST code and its application to the problems of canister cooling are described. This chapter also provides discussions of various TEMPEST models, from the base Version L4 to the advanced Version N29, which is currently in the development stage. Chapter 4 discusses the canister-filling experiments and the resulting data against which the TEMPEST predictions were compared. In Chapter 5, the TEMPEST input models used to simulate canister filling and cooling are described. Chapter 6 presents the results of the comparisons in both text and graphic form. These results are discussed and explained in Chapter 7. This chapter also provides detailed suggestions for future code development work that would make TEMPEST simulations more accurate, cost-effective, and convenient. In Chapter 8, quality assurance considerations related to TEMPEST development are presented. Input file information is included in Appendixes A through C.

# 2.0 CONCLUSIONS AND RECOMMENDATIONS

The experiments described in this report covered a wide range of scenarios typical of nuclear waste glass melter operations. Canisters of two sizes were instrumented, various types of insulation at various locations were tested, and both batch- and continuous-fill strategies were simulated. The conclusions and recommendations based on the results of this experiment are presented in this chapter.

## 2.1 CONCLUSIONS

The experimental data provided a sufficiently accurate basis for this code validation and will continue to be valuable for any future code validation study of glass pour and cooldown phenomena. The thermocouple (T/C) temperature readings were accurate to within 20°C or better, and the glass heights were accurately measured to within 1 in.

Generally, the TEMPEST-generated predictions were in good agreement with the experimental data, even though the radiation and front tracking models were quite crude and some uncertainty existed in the temperature of the initial glass pour. Version L4X of TEMPEST was basically a conduction-only model with a crude radiation correction. It was used for the batch-pouring scenario and assumed an instantaneous pour. Version M of TEMPEST incorporated the crude radiation correction but also tried to model the constantly rising glass surface. As might be expected, Version M predicted canister thermal history better than did Version L4X. There is a well-founded hope that Version N29, when completed, will provide temperatures accurate enough for complete engineering design specifications. Specific conclusions about the individual TEMPEST models are presented in the following paragraphs.

The TEMPEST batch-fill model, Version L4X, was sufficiently accurate to be useful for many LFCM applications, giving temperatures near the centerline that were accurate within a range of 50 to 125°C over most of the cooldown. However, during periods immediately following a pour, this TEMPEST model overpredicted temperatures by as much as 250°C.

These deviations were caused by a large underprediction of the cooldown immediately following a pour and by another, less severe, underprediction of the cooldown rate at times later than 10 h after a pour.

The Version L4X underprediction of the cooldown immediately following a pour was judged to be caused by two factors:

- Version L4X of TEMPEST apparently underpredicts the heat loss from the glass surface. This is due to the crude radiation model, the necessary temperature averaging, and the instantaneous batch-fill assumptions that allow no heat transfer during the filling process.
- Version L4X of TEMPEST also apparently underpredicts the radial and axial heat transfer just below the glass surface. This result is thought to be due to the TEMPEST code assumption that the glass is a nonconvective (conductive-only) medium. It could also be due to the use of a low thermal diffusivity at the very high glass temperatures in the canister.

The advanced TEMPEST Version N29 should eliminate both of these problems. Redoing the simulations with Version N29 should then result in much better agreement with experimental data.

The underprediction of the cooldown at times later than 10 h after a pour is probably the result of the canister insulation having about a 25% higher effective thermal conductivity than that used by TEMPEST. Further analysis is necessary to confirm this hypothesis.

The TEMPEST continuous-fill model, Version M, did a better job of predicting temperatures than did the batch-model. Temperatures predicted near the centerline were within 50°C or better over most of the cooldown. This was also true for temperatures at locations off the centerline for times after the glass surface passed a given axial level.

The continuous-fill model produced significant deviations in time (late predictions) and temperature (up to 400°C) for the hour or two after the glass surface passed a thermocouple level. However, the cooldown underpredictions were still better than those of the batch-fill model. The deviations are believed to be caused by a combination of the crude radiation model, the zero viscosity assumption used for the convecting glass, the poor resolution of heat transfer from the glass/air interface, and the lack of any heat transfer in the air above the glass. Again, use of advanced Version N29 should remedy this situation.

The turntable simulant experiments demonstrated that:

- The asymmetrical temperature profile caused by the asymmetrical turntable geometry can be modeled by TEMPEST.
- TEMPEST produces reasonable predictions of the temperatures within a turntable simulant, in spite of the code's limitations in computing simultaneous radiation and convective heat transfer, and in spite of considerable uncertainty in the emissivities of the stainless-steel surfaces.
- The best agreement between predicted and observed temperatures was found between the overpack to seal liner, the seal liner to thimble, and the thimble to turntable wall. Poorest agreement was observed between the canister wall to the overpack and the turntable wall to the ambient.

The validation would have been significantly improved, particularly for the turntable simulant experiment, if the experimental design had incorporated a heat balance. Additional improvements could have been made if there had been a direct means of measuring the glass temperature of the input pour stream directly at the point of impact with the glass surface. This would not only have helped specify initial conditions for TEMPEST, but would have provided valuable information about the heat loss from the stream of molten glass to the canister walls.

## 2.2 RECOMMENDATIONS

TEMPEST did an excellent job of qualitatively predicting the thermal response of the canister, and a fairly good job (subject to a few caveats) of predicting the quantitative behavior of the canister. The excellent qualitative agreement is due primarily to the kind of code TEMPEST is: a best-estimate code that solves the governing equations from fundamental physical principles--the conservation laws of mass, momentum, and energy. It provides the user with considerable latitude in selecting and refining a mesh to discretize a problem and, for codes of its size (a 30,000-line code representing more than 10 man-years of development effort), TEMPEST is very user-friendly.

Improved modeling in three main areas is needed for the base Version L4 of TEMPEST. These areas are

1. an implicit viscous diffusion model to simulate glass convection
2. an adaptive grid model capable of tracking the moving front of the air/glass interface
3. a graybody enclosure radiation model that will work efficiently in conjunction with the current convection solution algorithm.

We therefore recommend that these superior models be installed into a new advanced version of TEMPEST. The N29 version of TEMPEST currently under development already has some of the above-mentioned features installed. Specifically, the implicit viscous diffusion model has already been installed and successfully tested; the N29 version of TEMPEST will allow the efficient computation of the convection of any fluid, regardless of how high the fluid viscosity is.

The N29 version of TEMPEST is already well along the path to becoming a best-estimate code for modeling canister pouring and cooldown. We recommend that the development effort on TEMPEST Version N29 be continued according to the following agenda:

1. Complete the front tracker model; install, test, and debug.
2. Carryover the crude radiation model from Versions L4X and M into Version N29.
3. Test the resulting code on one canister run, and assess the code accuracy.
4. If the code accuracy is acceptable, rerun the remaining cases.

5. If the code accuracy is unacceptable, install the general graybody radiation model; then test the resulting code on one canister run and assess the accuracy.

6. If the code accuracy is acceptable, rerun the remaining cases.

7. Document the final advanced version of the code and publish it.

In addition to improvements with TEMPEST, several features of the glass pouring and canister cooldown that were uncovered by the PSCM runs merit further attention. Specific recommendations include:

- Establish the actual thermal properties (thermal conductivity, specific heat, and density), over the entire temperature range, up to 1150°C, for the waste glasses described in this report.
- Establish, through modeling and further experimentation, the temperatures of the glass stream when it impacts the glass surface inside the canister.
- Establish the emissivity of the stainless-steel canisters and the speed with which the canisters oxidize and change. This is a major source of uncertainty in the modeling efforts.
- Establish the actual conductivities of the Durablanket and Fiberfrax insulation, especially in the semi-crushed state. If the actual conductivities are about 25% lower than the manufacturer-stated value, the TEMPEST predictions would be more accurate.
- Improve the accuracy in determining the position of the glass/air interface with time. This would be accomplished by using improved methods of measuring canister weight and glass volumetric flow.
- Improve the understanding of the thermocouple response to the rising glass surface just before and after the glass surface covers the thermocouples.

# 3.0 TEMPEST CODE DESCRIPTION

The TEMPEST code was chosen for modeling glass temperatures during canister filling and cooldown. TEMPEST is a transient, three-dimensional, thermofluid computer program developed by PNL to analyze a broad range of fast transient, coupled fluid dynamic, and heat transfer systems. (Trent, Eyler, and Budden 1983[a]; Trent and Eyler 1985; Eyler and Trent 1984.) The code solves the full three-dimensional, time-dependent equations of motion, continuity, and heat transport for either laminar or turbulent fluid flow (including heat diffusion and generation) in both solids and liquids. TEMPEST provides finite difference solutions to these equations in either cylindrical or cartesian coordinates for incompressible flows with small density variations. The capabilities, limitations, and versions of the TEMPEST code are described in this chapter.

## 3.1 CAPABILITIES AND FEATURES

The TEMPEST code was selected for use in this study for five primary reasons:

- TEMPEST can compute combined conduction and convection directly based on fundamental physical principles (the physical conservation laws) without having to use empirical convection-related heat transfer coefficients.
- TEMPEST allows for three-dimensional solutions of transient thermofluid phenomena using temperature-dependent material properties that can be specified by the user. TEMPEST can model density-induced flows and the large changes in viscosity with temperature that are typical of glasses.

(a) Trent, D. S., L. L. Eyler, and M. J. Budden. 1983 TEMPEST - A Three-Dimensional Time-Dependent Computer Program for Hydrothermal Analysis. Volume I: Numerical Methods and Input Instructions. Pacific Northwest Laboratory, Richland, Washington.

- TEMPEST is accurate and user-friendly. Significant verification and bench-marking experience with the code has been accumulated and documented (Eyler, Trent, and Budden 1983).[a]
- TEMPEST is portable; it is written in ANSI FORTRAN-77 and will run on virtually any machine. It is currently being used on CDC-7600 computers, as well as CRAY-1S, CRAY-XMP, DEC-VAX, and IBM machines.
- TEMPEST is available for continued code development and refinement, so that discrepancies in the comparison of any results with computer predictions can be rectified.

## 3.2 CODE LIMITATIONS

Despite its advantages, the base version of TEMPEST, designated L4, is not entirely without weaknesses when modeling the complex thermofluid phenomena associated with pouring waste glass into canisters and the following cooldown.

The original TEMPEST code was developed primarily to provide a user-oriented thermofluid analysis tool capable of simulating a wide range of fast breeder reactor component flow problems. Such flow problems involve fast transients where conduction and convection are the primary heat transfer mechanisms (except possibly at the boundaries). Because sodium is opaque to thermal radiation, a general graybody enclosure radiation model is not necessary.

However, the glass pouring and cooldown procedures result in relatively slow transients in which radiation is also a dominant mode of heat transfer. Inside the glass, conduction and, at high temperatures, convection are also important. The large temperature changes result in extremely wide ranges of viscosity that TEMPEST was not designed to handle.

Related to potential cracking and devitrification is the cooldown history of a canister when emplaced on a turntable with other canisters in various

[a] Eyler, L. L., A. S. Trent, and M. J. Budden. 1983. TEMPEST - A Three-Dimensional Time-Dependent Computer Program for Hydrothermal Analysis. Volume II. Assessment and Verification Results. Pacific Northwest Laboratory, Richland, Washington.

stages of fill/cooldown themselves. Modeling the cooldown history requires a nonconcentric cylindrical coordinate system, or even a multi-region coordinate system. TEMPEST does not currently have such a capability.

Finally, TEMPEST, like most codes, is a fixed grid code. Pouring glass into the canisters causes a moving boundary or interface between glass and air. For best accuracy, the moving glass/air interface must also be modeled.

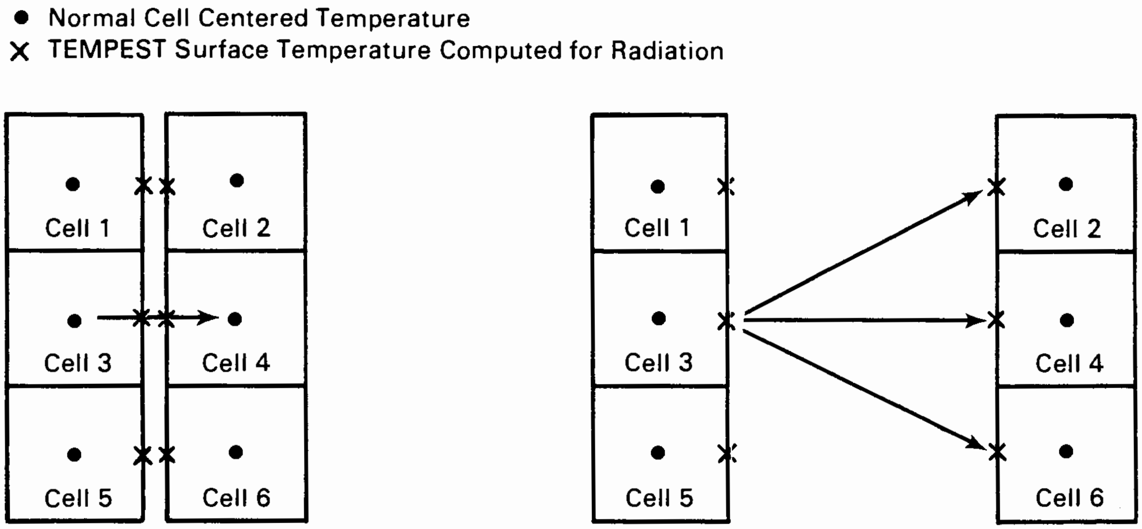
For these reasons, the actual TEMPEST models described in this report contain simplifications and approximations, some of which are significant. These limitations, approximations, and modeling procedures are explained in the following four sections.

### 3.2.1 Radiation Model

A general graybody enclosure radiation model is needed if TEMPEST is to be used to effectively address the glass-pouring problem. Currently, the base version of TEMPEST (Version L4) has only a gap radiation model (see Figure 3.1a). With the gap radiation model, TEMPEST calculates the cell edge temperatures for use in the Stefan-Boltzmann law, implicitly assuming that the shape factor between any two cells separated by a radiation connector is 1. For example, in Figure 3.1a this means that radiant energy flows from cell 3 to cell 4 only. In reality, energy flows as depicted in Figure 3.1b, where cell 3 radiates to cells 2, 4, and 6, each with a different shape factor.

The TEMPEST gap radiation model is sufficient for use as an external boundary condition or in the interior of a region across a truly narrow gap where the details of what goes on in the gap (e.g., convection) are not important. However, it is not appropriate for modeling the heat transfer from the glass surface to the canister side walls or out the top of the canister. The glass surface/canister wall geometry for which a radiation model is needed is shown in Figure 3.2.

To improve on the radiation model available in Version L4 of TEMPEST, a new version, designated L4X, was built to better approximate the thermal phenomena in the canisters. A crude radiation model was included in this version. In this model, the average glass surface temperature was calculated, and the canister wall temperatures above the glass surface were calculated.


● Normal Cell Centered Temperature
× TEMPEST Surface Temperature Computed for Radiation
Cell 1
Cell 2
Cell 3
Cell 4
Cell 5
Cell 6
Cell 1
Cell 2
Cell 3
Cell 4
Cell 5
Cell 6

a. Gap Radiation Model

b. Enclosure Model

FIGURE 3.1. Gap Radiation Approximation Compared to Enclosure Radiation Model

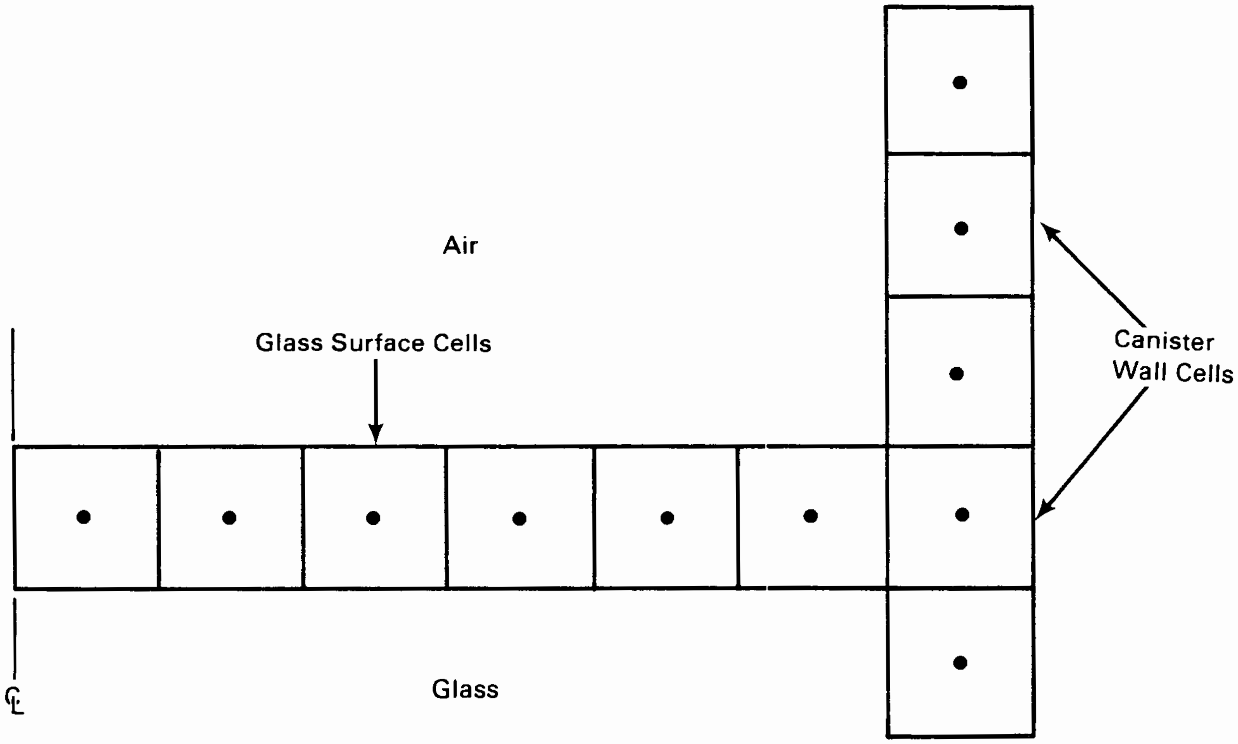


FIGURE 3.2. Geometry for Radiation from Glass Surface Cells to Canister Wall Cells

These temperatures were used to calculate the thermal energy leaving the glass surface. The canister wall temperature was evaluated in two ways, designated HI and LO.

For the HI model, the average canister wall temperature was taken to be the temperature of the canister wall cell at the same height as the glass surface. Because the canister surface above the glass is cooler than the canister surface at the glass contact, this approximation probably overestimates the average canister temperature, which underestimates the radiant loss from the glass surface.

For the LO model, the average canister temperature was assumed to be the arithmetic average between the canister wall temperature at the glass surface level and the canister wall temperature at the top of the canister. This should generally underestimate the average canister temperature and, thus, overestimate the heat loss from the glass surfaces.

For both models, the calculated radiant energy lost from the glass surface cells was subtracted from those cells, but not added elsewhere. Stability reasons precluded adding that energy to the canister walls. It was thought that the HI and LO models would bracket the experimental results.

A more exact treatment of thermal radiation is needed. The gap radiation model of the L4 version is not adequate, and the crude approximation installed in the interim M version, although qualitatively correct, should be made more exact and flexible.

### 3.2.2 Transient Duration

The original TEMPEST Version L4 is best able to handle transients that are "fast," on the order of seconds. Because of the explicit nature of the numerical algorithm for momentum convection, relatively small time steps are required if fluid flow is to be simulated. However, the canister filling and cooldown transients are much slower, lasting several hours. This period is caused by the long pour time and the large thermal inertia of the glass and canister combination. Furthermore, the allowable time-step size for the glass pouring problem is much smaller than it would be if a different fluid (e.g., water, sodium, or air) were being used. This difference is due to the inverse relationship between time-step size and fluid viscosity typical of any explicit convection scheme (Roache 1976). Therefore, Version L4 is not suited for modeling glass flow where the flow field needs to be calculated. On the other hand, the heat transfer solution in TEMPEST is implicit, and heat-transfer-only simulations can be calculated efficiently by using large time steps.

### 3.2.3 Geometries

Currently, TEMPEST is able to handle either cartesian or cylindrical geometries. It is not able to handle mixed geometries (a problem where some regions are cartesian and some regions are cylindrical) or nonconcentric cylindrical geometries.

Nonconcentric cylindrical geometries are of interest to the glass-pouring and cooldown problem because of the geometry of most melter turntables, where the canisters are rotated from underneath the melter and allowed to cool. A top view of the canisters in a typical turntable arrangement is shown in Figure 3.3. The capability to model this arrangement is called a multiple-grid or body-fitted coordinate system. The L4 version of TEMPEST has neither capability.

### 3.2.4 Grid Boundaries (Glass/Air Interface)

The glass-pouring problem requires that the moving boundary of the glass surface be tracked as it rises in the canister. During the pour, hot glass streams down the middle of the canister and impacts the surface of the glass. From there it both spreads along the surface and penetrates the molten glass below the surface. The degree to which the glass penetrates or spreads along the surface is controlled by the initial momentum of the glass stream and the local viscosity, which is highly temperature-dependent. Below the glass surface, molten glass convects and slowly congeals. Above the glass surface, air convects heat away from the glass and canister and out the top.

During the pour, the actual glass surface moves, resulting in a moving boundary problem. To handle a range of glass-pouring problems, a code would need to handle an arbitrary pour scenario that includes stopping and starting. TEMPEST does have a flexible time-dependent table feature (user-supplied at input), but not a moving boundary capability.

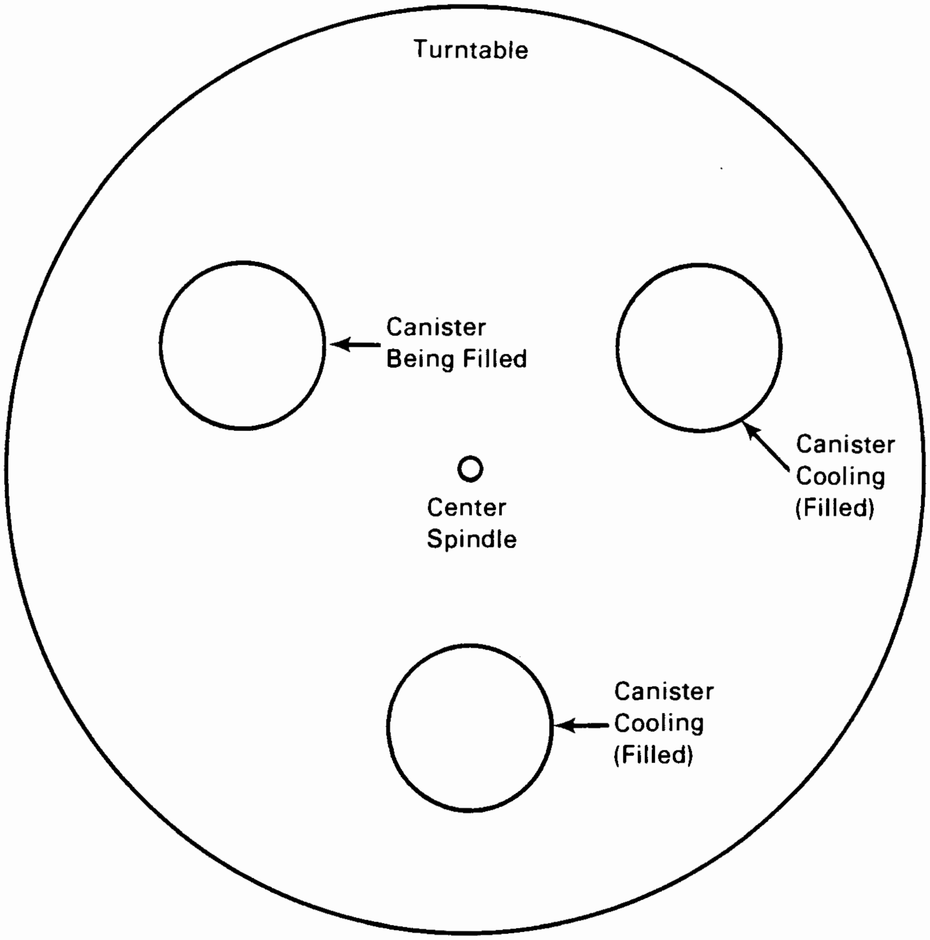


FIGURE 3.3. Canister and Turntable Arrangement

## 3.3 CODE VERSIONS

TEMPEST is a living code. The released version, designated L4, is available from the National Energy Software Center, Argonne, Illinois. In addition, several advanced versions are available at PNL for various research problems. These advanced versions are constantly undergoing change as new problems arise that need analysis, but which are not immediately amenable to solution by TEMPEST.

Version L4 of the code was not powerful enough to adequately analyze the glass-pouring and cooldown problem. An experimental version, L4X, was created to add the LO and HI radiation capability described in Section 3.2.1. However, glass convection and a moving boundary capability were not available with the L4X version.

To add glass convection and the moving boundary capability, another version was created, designated M. This version incorporated the LO and HI radiation capability of L4X, but in a very restrictive, nongeneral way. Version M was created quickly, to assess the increase in accuracy that could be obtained by simulating convecting glass and the moving glass/air interface. This TEMPEST version crudely tracked the glass interface, but allowed no convection of air above the glass surface. It was designed to handle both batch- and continuous-fill scenarios.

The results of the simulations with Versions L4X and M, coupled with the importance of accurate glass thermal history predictions, led to initial development of a better version that would address all of the problems discussed in Section 3.2. This version is currently designated N29. Version N29 is still under development and was not used to rerun the predictions.

Version L4X of the code was used exclusively for the batch-fill scenarios. Version M was used for canisters modeled with a continuous (time-varying) pour. The turntable simulant, in which a canister was insulated asymmetrically around its circumference to simulate the nonconcentric geometry of a turntable, was also run with Version L4X.

# 4.0 EXPERIMENTAL DATA COLLECTION

The data against which TEMPEST-generated predictions were compared were collected during two experiments with the pilot-scale ceramic melter (PSCM) at PNL. The melter, described in Chikalla and Powell (1982), is a source of molten, nonradioactive glass at about 1150°C. This chapter provides a description of the equipment and methodology used to generate the data.

## 4.1 CANISTER DESCRIPTION

Five canisters of three different designs were used in the two PSCM experiments. The canister design features are compared in Table 4.1. Three canisters had 12-in. nominal diameters; the other two were 13 in. and 24 in. in diameter. All canisters were made of 304L stainless steel. Canister height varied from 45.4 in. to 85.4 in. The bottoms of the 12-in.-diameter canisters were flat except for a 0.5-in. lip and a 3.5-in.-diameter by 1.6-in.-high cylindrical bulge in their centers (see Figure 4.1). The bottom of the 13-in.-diameter canister was slightly rounded and supported by four bars located at 90° intervals beneath the canister. The 24-in.-diameter canister's bottom was dished up 1.06 in. in the center, in a reversed flange design. The radial variation of this design is shown in Table 4.2.

The inside surfaces of the 13-in.-diameter canister were lined with 1/8-in.-thick Fiberfrax[a] paper and a graphite mold release compound. Half of the bottom and a 180° arc of the side were lined with Fiberfrax; the remainder of the canister was coated with graphite mold release agent. This was done to compare the effects of Fiberfrax, which minimizes both shear and compressive stresses, to the mold release compound, which reduces only the shear stresses at the wall. This canister was insulated with the same total amount of combined internal and external insulation in both the graphite and the Fiberfrax sides, to keep radial temperature differences the same for both the

[a] Fiberfrax is a ceramic paper manufactured by Carborundum Resistant Materials Co., Insulation Division, Niagara Falls, New York.

TABLE 4.1. Canister Information

| | Canister | | | | |
|---|---|---|---|---|---|
| Design Features | 12LI | 12HI | 12TTS | 13 | 24 |
| PSCM run | 20 | 20 | 20 | 21 | 21 |
| Height, in. | 45.4 | 45.4 | 45.4 | 56.0 | 85.0 |
| Outside diameter, in. | 11.75 | 11.75 | 11.75 | 12.75 | 24.0 |
| Inside diameter, in. | 11.12 | 11.12 | 11.12 | 12.25 | 23.5 |
| Wall thickness, in. | 0.31 | 0.31 | 0.31 | 0.25 | 0.25 |
| Liner | | | | | |
| Material | FF[a] | FF | FF | FF&gr[b] | Dura[c] & FF |
| Thickness, in. | 1/8 | 1/8 | 1/8 | 1/8[b] | 1/4 sides<br>1/2 bot |
| External insulation | | | | | |
| Material | Dura | Dura | Dura | Dura & FF[d] | None |
| Thickness, in. | 1/4 | 3 | -- | 1/4 & 3/8[d] | -- |
| Bottom design | bulge | bulge | bulge | rounded | dished |

(a) Fiberfrax ceramic paper type 970J (12 lb/cu. ft. density).
(b) A 180° arc and half of the bottom of this canister were lined with 1/8 in. of Fiberfrax; the remainder of the canister was coated with a graphite mold release compound (Dylon Grade AE Mold Release Agent, Dylon Industries).
(c) Durablanket (8 lb/cu. ft. density).
(d) A 180° arc and half of the canister bottom were externally insulated with 1/8 in. of Fiberfrax and 1/4 in. of Durablanket (graphite side, see (b)); the remainder of the canister was externally insulated with only 1/4 in. of Durablanket.

graphite- and Fiberfrax-lined regions. Thus, in addition to the 1/4 in. of Durablanket[a] placed over the entire surface, 1/8 in. of Fiberfrax paper was used on the outside of the graphite-coated region. The other canisters were covered with 0 in. to 3 in. of Durablanket insulation (Table 4.1).

[a] Durablanket is a ceramic fiber blanket. It is manufactured by Carbonumdum Resistant Materials Co., Insulation Division.

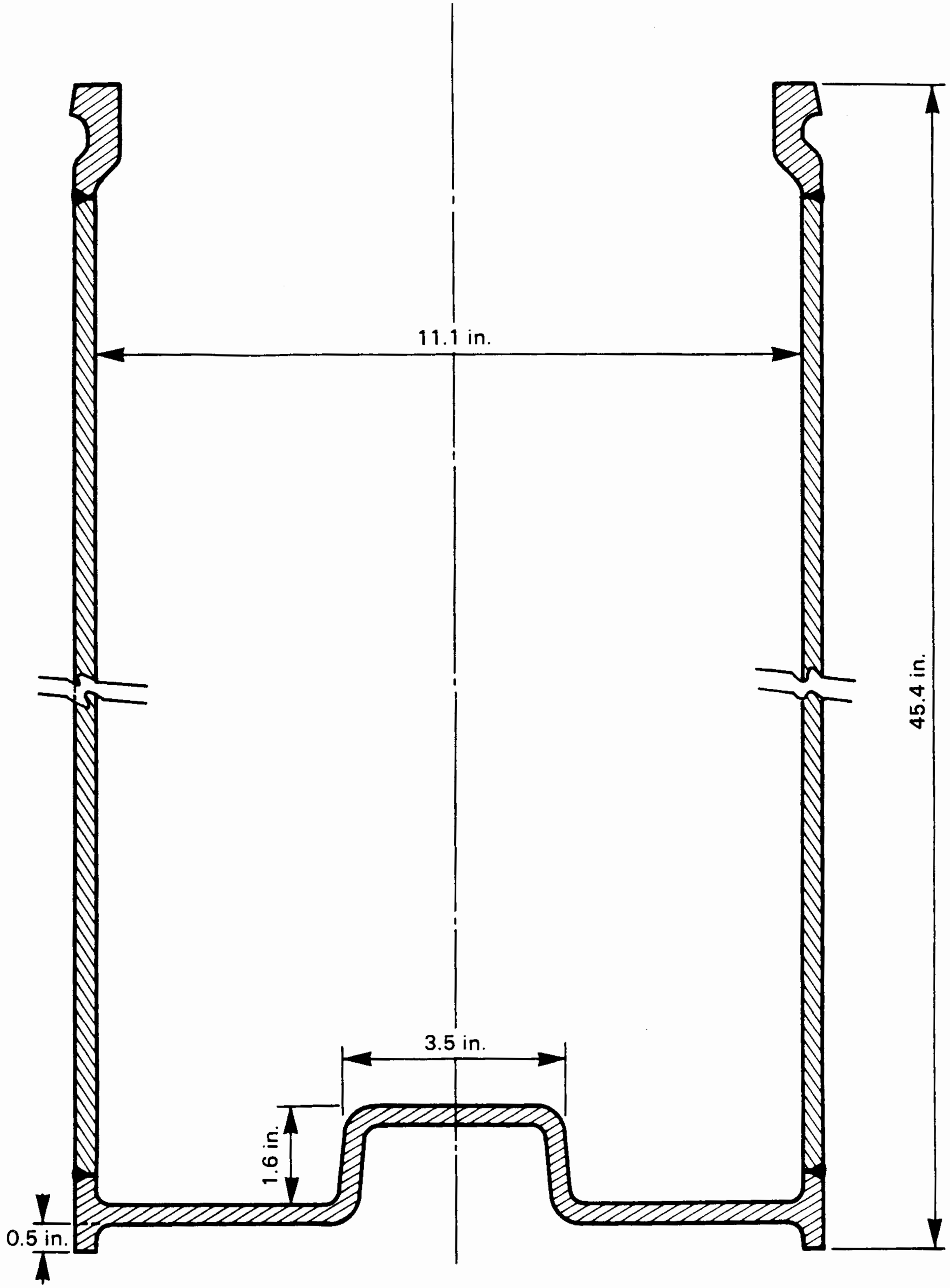


FIGURE 4.1. Cross Section of 12-in.-Diameter Canisters

TABLE 4.2. Measured Distances from the Bottom of the Reverse Flanged 24-in.-Diameter Canister

| Distance from Center, in. | Distance Above Outside Edge, in. |
|---|---|
| 0 | 1.06 |
| 1.5 | 0.89 |
| 3.0 | 0.63 |
| 4.5 | 0.50 |
| 6.0 | 0.38 |
| 7.5 | 0.25 |
| 9.0 | 0.13 |
| 10.5 | 0.06 |

The filling arrangement used for the 13-in.-diameter canister is shown in Figure 4.2. The PSCM-to-canister adapter was inserted about 6 in. into the canister. An air seal to prevent unwanted air from entering the PSCM was formed by inserting a 4-in.-wide strip of Durablanket into the annulus. This type of seal was used in an attempt to minimize vertical forces between the adapter and the canister; this allowed for accurate weight readings to be obtained by the load cell.

The arrangement shown in Figure 4.2 is identical to that used for two of the three 12-in.-diameter canisters (12LI and 12HI--see Table 4.1), except that the PSCM-to-canister adapter was placed about 2 in. above the canister. An air seal was formed by wrapping Durablanket around the upper portion of the canister and the lower portion of the adaptor and securing it with a roll of sheet metal held in place by hose clamps.

## 4.2 TURNTABLE SIMULANT

The radioactive liquid-fed ceramic melter (RLFCM) at PNL is equipped with a turntable that can hold up to three canisters at one time. A computer model is needed to compute the heat transfer within such turntables. To determine the ability of the TEMPEST code to compute the heat transfer within nonconcentric, multi-barrier geometries, the third 12-in. canister (canister 12TTS--see Table 4.1) was filled while within a turntable simulant (TTS). The purpose of

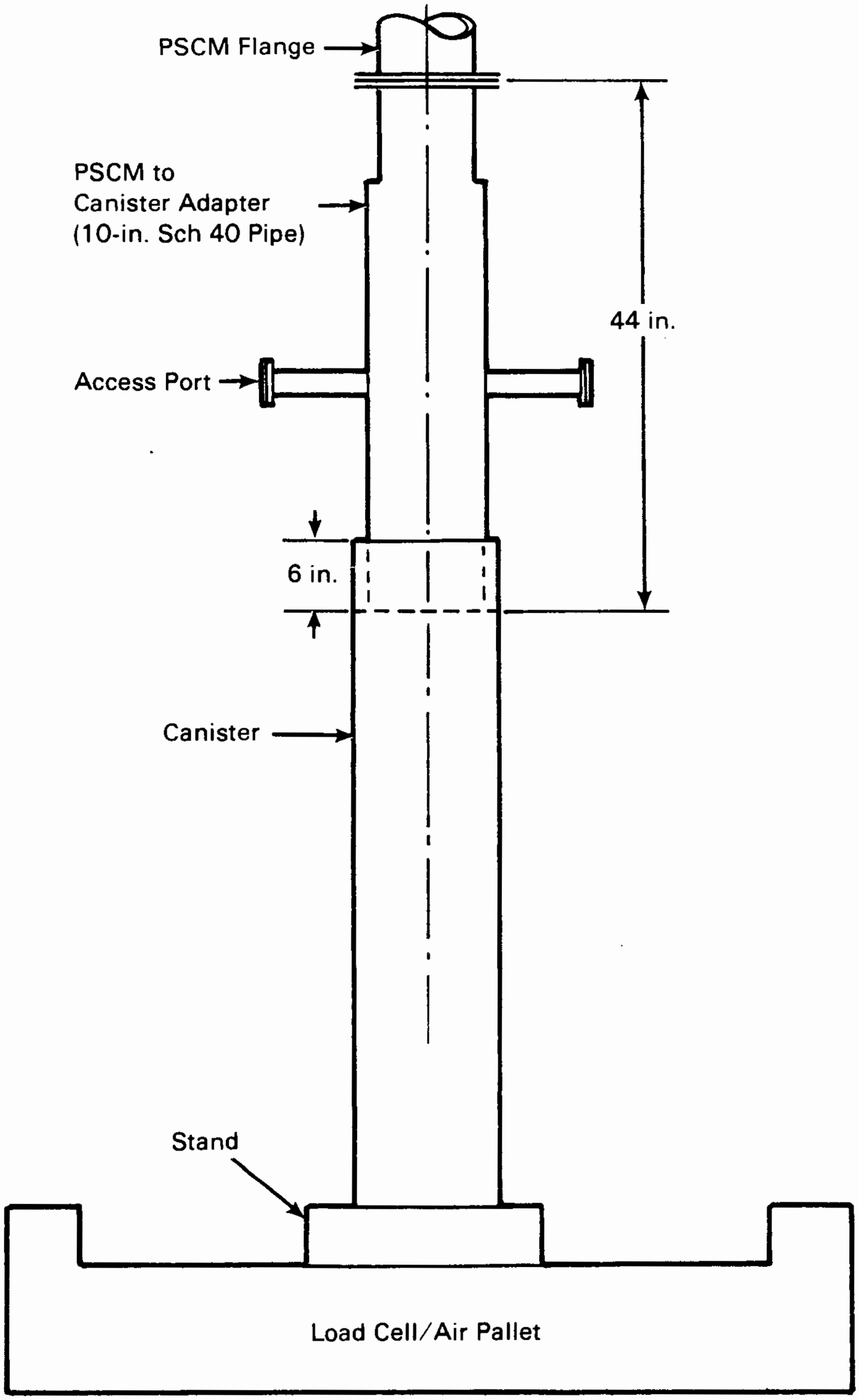


FIGURE 4.2. Canister Placement Under Melter for 13-in.-Diameter Canister

the TTS was to mimic, although not duplicate, the heat transfer resistances found in the RLFCM turntable at PNL. The TTS consisted of three concentric shells located off-center in a 46.5-in.-diameter shell (Figure 4.3). In order of increasing distance from the canister, these shells are referred to as the overpack, the liner, the thimble, and the turntable. The shells have outside diameters of 14 in., 18 in., 24 in., and 46 in., respectively. All shell surfaces within the thimble were concentric; however, the thimble shell was positioned off-center in the turntable shell such that its point of closest approach to the turntable was 1 in.

The tops of the thimble and liner were 1.5 in. below the top of the TTS shell. The tops of the canister and the overpack were 5.25 in. below the top of the liner. The canister was held 1 in. above the bottom of the overpack by crossbars. Concentricity of the canister in the overpack was ensured by four 1-in. bars, located at 90° intervals, that ran vertically the length of the canister. These vertical bars limited air flow between the quadrants formed by the bars.

The overpack was also held within the liner by crossbars that permitted essentially free flow of air between the large cavity under the overpack and the annulus around the canister. Approximately 2 in. of Durablanket were placed at the top between the liner and the thimble, to prevent air flow from this annulus to the remainder of the TTS.

The 24-in.-diameter canister was also filled while in the TTS, to permit a measure of the amount of material volatilized from the glass. However, all inside apparatus, i.e., the overpack, liner, and thimble, were removed, and the canister was placed at the center of the 46-in.-diameter shell.

The lid of the TTS was bolted to the PSCM flange. A water seal was used to minimize vertical forces between the TTS lid and the remainder of the TTS. The TTS, in turn, rested upon a load cell; the weight of the glass that was added was determined by the indicated changes in weight.

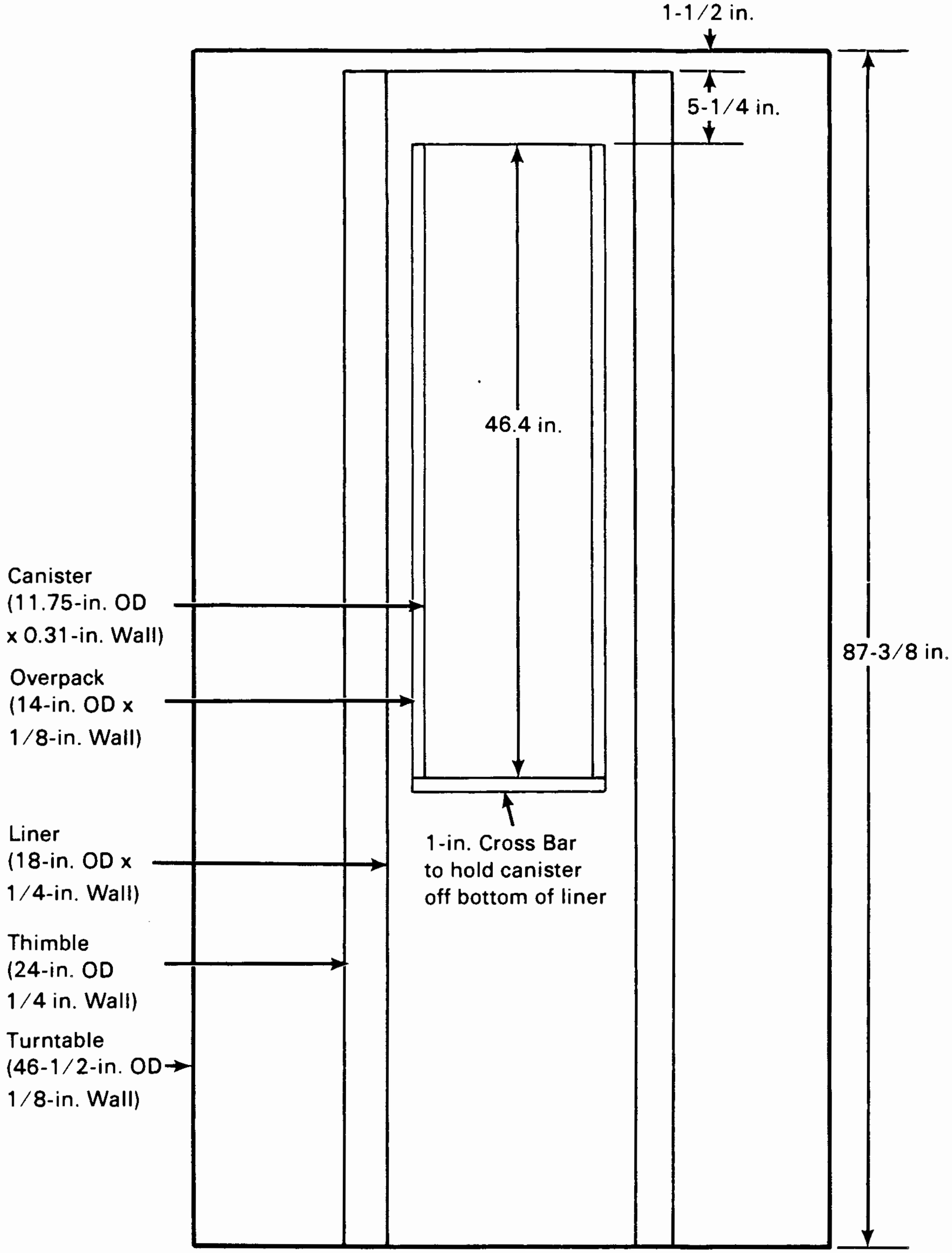


FIGURE 4.3. Turntable Simulant Model Configuration

## 4.3 THERMOCOUPLE LOCATIONS

Stainless-steel sheathed type K thermocouples, 1/16 in. and 1/8 in. in diameter, were used to measure canister temperatures, glass temperatures and temperatures within the TTS. The thermocouple outputs were fed into a datalogger and recorded on paper and magnetic tape. The data were manipulated by digital computers.

Calibration checks at 300°C, 600°C, and 900°C on a representative sample (57%) of the thermocouples gave a maximum error of 7°C. Computations made with the TEMPEST code also yielded thermocouple conduction errors ranging from -0.8°C to -22.7°C (see Tables 4.3 and 4.4). These errors depended upon glass temperature, thermocouple diameter, and depth of immersion.

Thermocouples were placed on the outside surfaces of all five canisters. These externally-mounted thermocouples were placed in contact for 4 in. horizontally along the wall and were coated with 1/8 in. of Ceramobond 571[a] cement to improve the thermal contact between the thermocouple junction and the wall of the canister. In addition, thermocouples were located within all but canister 12TTS. The thermocouple locations in the canister glass were selected to give data on the radial temperature distributions within the canister; thermocouples were located at vertical positions to give information on thermal conditions as a function of distance from the glass/air interfaces.

Figures 4.4 and 4.5 give the 30 thermocouple locations for canisters 12LI and 12HI. As shown, one thermocouple was located 5 in. from the edge on the bottom of each canister; additional internal thermocouples were emplaced at eight axial locations. Internal thermocouples placed 1, 2, and 4-1/4 in. from the inside wall were located at all levels per Figure 4.5, except for level 7, which had only a 4-1/4-in. thermocouple.

Thermocouples were not placed in the center of the canisters, to minimize interactions with the glass stream. Nevertheless, radial migrations during glass pouring caused by a deteriorated pour tip resulted in some glass

[a] Ceramobond is a ceramic cementing agent manufactured by Aremco Products, Ossining, New York.

TABLE 4.3. TEMPEST-Determined Thermocouple Conduction Factors for Canisters with 1/4-in. Durablanket Insulation

| Thermocouple Diameter, in. | Thermocouple Distance from Canister Inside Wall, in. | Conduction Error at Temperature Within TTS, °C | | |
|---|---|---|---|---|
| | | 300°C | 600°C | 1100°C |
| 1/16 | 1 | -4.1[a] | -6.3 | -19.8 |
| 1/8 | 2 | -5.4 | -8.1 | -22.7 |
| 1/8 | 4.25 | -4.0 | -7.3 | -11.0 |

(a) A negative sign signifies that the thermocouple is giving a low value.

TABLE 4.4. TEMPEST Determined Thermocouple Conduction Factors for Canisters with 3-in. Durablanket Insulation

| Thermocouple Diameter, in. | Thermocouple Distance from Canister Inside Wall, in. | Conduction Error at Temperature Within TTS, °C | | |
|---|---|---|---|---|
| | | 300°C | 600°C | 1100°C |
| 1/16 | 1 | -0.9[a] | -1.6 | -5.0 |
| 1/8 | 2 | -1.2 | -2.3 | -5.0 |
| 1/8 | 4.25 | -0.8 | -1.9 | -3.0 |

(a) A negative sign signifies that the thermocouple is giving a low value.

splattering on the thermocouples located 2 in. and 4-1/4 in. from the canister wall. However, the diameter and support structure for thermocouples (see Figure 4.5) were selected so that the thermocouples could remain their positions in the canister.

The 30 thermocouple locations for the 13-in.-diameter canister are given in Table 4.5. Six levels were used; five thermocouples were located at each level as follows:

- on the outside surface of the canister (1/16-in. diameter)
- on the inside surface of the Fiberfrax liner (1/16-in. diameter)
- 1 in. from the canister inside wall (1/16-in. diameter)

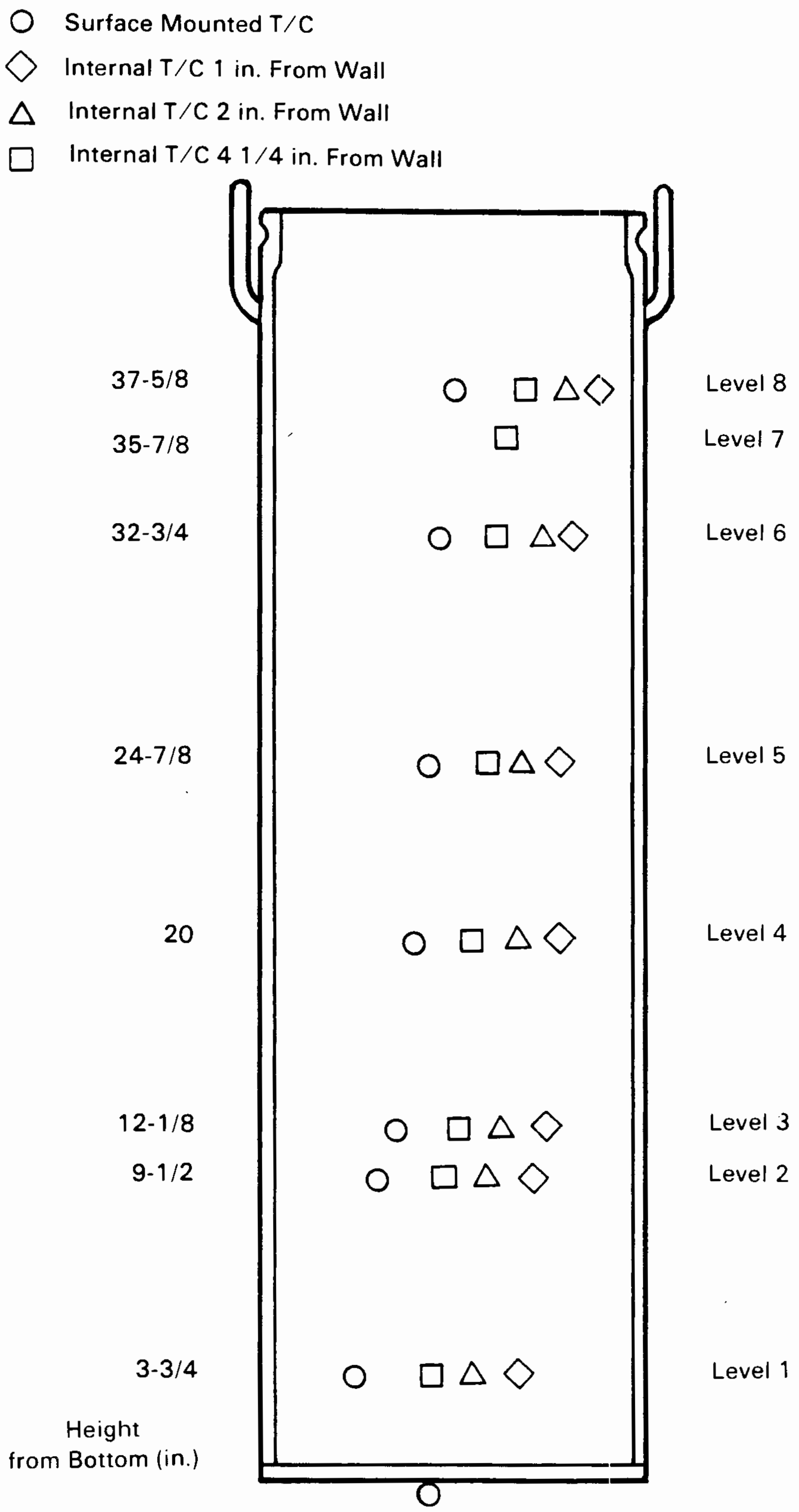


FIGURE 4.4. Thermocouple Locations for Canisters 12LI and 12HI

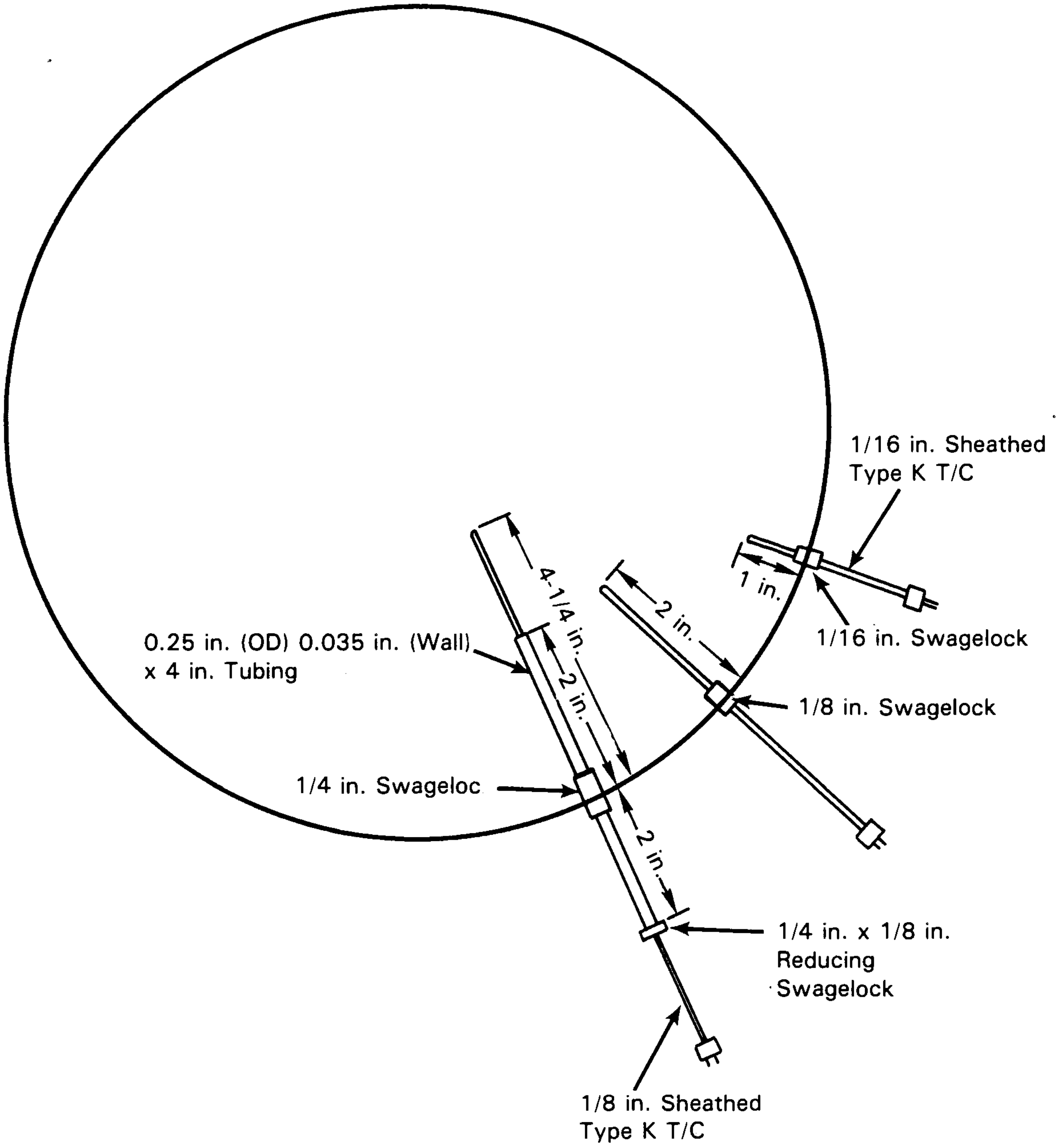


FIGURE 4.5. Typical Placement and Structural Support of Internal Thermocouples on Canisters 12LI and 12HI

TABLE 4.5. Axial Thermocouple Locations for the 13-in.-Diameter Canister

| Level | Distance from Outside Bottom of Canister, in. |
|---|---|
| 1 | 3.5 |
| 2 | 13.3 |
| 3 | 21.2 |
| 4 | 26.1 |
| 5 | 34.0 |
| 6 | 38.4 |

TABLE 4.6. Axial Thermocouple Locations for the 24-in.-Diameter Canister

| Level | Distance from Outside Bottom of Canister, in. |
|---|---|
| 1 | 2.25 |
| 2 | 5.25 |
| 3 | 14.25 |
| 4 | 21.75 |
| 5 | 29.25 |
| 6 | 36.75 |
| 7 | 44.25 |
| 8 | 53.25 |
| 9 | 62.25 |
| 10 | 65.25 |

- 2 in. from the canister inside wall (1/8-in. diameter)
- 4-1/4 in. from the canister inside wall (1/8-in. diameter).

The thermocouple locations for the 24-in.-diameter canister are given in Table 4.6. Five thermocouples were located at each of ten levels. The thermocouple locations were:

- on the outside surface of the canister (1/16-in. diameter)
- on the inside surface of the 1/4-in.-thick Fiberfrax liner (1/16-in. diameter)
- 2.5 in. from the canister inside wall (1/16-in. diameter)
- 5 in. from the canister inside wall (1/8-in. diameter)
- 10 in. from the canister inside wall (1/8-in. diameter).

The innermost three thermocouples were supported similarly to those shown in Figure 4.5, except that:

- The 2.5-in. thermocouples were supported by a 1/8-in.-diameter tube extending 2 in. from the wall.
- The 5-in. thermocouples were supported by a 1/4-in.-diameter tube extending 2.5 in. from the wall.
- The 10-in. thermocouples were supported by a 1/4-in.-diameter tube extending 8 in. from the wall.

The primary purpose of the experiment in which canister 12TTS was filled while in the TTS was to determine the heat transfer behavior external to the canister. Because of this, and because of the space limitations arising from the small clearance involved, thermocouples were not placed inside this canister. Instead, details on the three-dimensional surface temperature profiles in the TTS were emphasized. Thermocouples were placed at 3 circumferential and 20 axial locations on the canister. The circumferential locations corresponded to the side closest to the outer TTS wall, farthest from the outer TTS wall, and midway (90°) between these two radial positions. Additional thermocouples were placed 1) at adjacent radial and axial locations on the overpack, liner, thimble, and turntable walls at the above-mentioned 20 levels, and 2) at additional axial levels above and below the canister.

## 4.4 CANISTER FILLING METHODS

Two methods were used to fill the canisters with molten glass: continuous filling and batch filling. Batch filling is necessary when the glass production rate from the melter is too slow to adequately fill a canister continuously. If the fill rate is too slow, voids may be formed because the glass cools too quickly to flow to the outside edge of the canister. Also, the glass may pillar in the canister, necessitating canister removal before filling is complete. The three 12-in.-diameter canisters from the PSCM-20 experiment were filled in three batch pours of about 53 kg each; the 24-in.-diameter canister used in the PSCM-21 experiment was filled in 33 batches ranging from about 10 kg to 60 kg each, as part of a study of the effect of batch size on product quality, e.g., cracking and voids. The 13-in.-diameter canister from PSCM-21 was filled continuously at a nominal rate of 0.8 kg/min, for a total glass mass of 207 kg over 260 min.

## 4.5 GLASS MEASUREMENTS

Information required for TEMPEST validation included the amount of glass and the location of the glass surface after each glass pour. Three methods were used to obtain this information:

- weight measurements
- direct measurements of the surface by lowering a weight (sinker) to the surface
- observation of thermocouple response, to infer when surface coverage occurred.

Interactions between the canister and the PSCM-to-canister adapter significantly reduced the usefulness of the weight measurements. In addition, procedural difficulties resulted in some loss of glass height data taken by use of the sinker. Thermocouple response, combined with sinker measurements, proved to be most useful in determining glass height during the filling sequence.

The glass surface in the canister is not perfectly flat, either during filling or after cooling. It is not uncommon for a cold filled and cooled canister to develop a sinkhole up to a few inches in diameter and depth at the center of the surface. In addition, glass beads and cylinders of perhaps 1/2 in. in diameter may be found on the surface. These are formed when the glass pour stream is nearly finished and has therefore cooled significantly.

Significant irregularities in the hot surface as the glass pours onto it have been observed during other PSCM experiments. One such irregularity is lobing, where the glass moves out from the center in lobes, first covering one sector, then another. These irregularities are minimized at the nominal 2-kg/min pour rates used during batch pours, but could be significant at the 0.8-kg/min average pour rate for the 13-in.-diameter canister.

A reasonable value for accuracy of the measured glass heights is about 1 in. during the pours and about 1/2 in. for the final height, because the latter could be checked after the canister had cooled. Times of coverage are probably accurate to 6 min or better.

The glass heights as a function of elapsed time from start of fill for canisters 12LI, 12HI, and the 13-in.-diameter canister are given in Tables 4.7, 4.8, and 4.9, respectively. The long time between the first and second pours of canister 12HI resulted from a melter experiment in which the pour was interrupted for a day to simulate an LFCM process upset. The data for for 24-in.-diameter canister are presented in a somewhat different format in Table 4.10 because the TEMPEST simulation treated this case as a continuous fill of varying filling rate, even though this canister was batch-filled. This approach significantly reduced the input modeling effort.

TABLE 4.7. Glass Height for Canister 12LI

| Pour Number | Elapsed Time, min | | Final Glass Height Measured from Outside Bottom of Canister, in. |
|---|---|---|---|
| | Start | Finish | |
| 1 | 0 | 23 | 12.9 |
| 2 | 304 | 360 | 25.7 |
| 3 | 714 | 763 | 39.5 |

TABLE 4.8. Glass Height for Canister 12HI

| Pour Number | Elapsed Time, min<br>Start | <br>Finish | Final Glass Height Measured from Outside Bottom of Canister, in. |
|---|---|---|---|
| 1 | 0 | 27 | 13.1 |
| 2 | 1758 | 1797 | 27.0 |
| 3 | 2037 | 2089 | 38.5 |

TABLE 4.9. Glass Height for the 13-in.-Diameter Canister

| Elapsed Time, min | Final Glass Height Measured from Outside Bottom of Canister, in. |
|---|---|
| 26 | 3.5 |
| 86 | 13.3 |
| 155 | 21.2 |
| 170 | 26.1 |
| 224 | 34.0 |
| 254 | 38.4 |
| 263 | 40.1 |

TABLE 4.10. Glass Height for the 24-in. Diameter Canister

| Pour Number | Elapsed Time, min |  | Final Glass Height Measured from Outside Bottom of Canister, in. |
|---|---|---|---|
|  | Start | Finish |  |
| 1 | 0 | 24 | 4.3 |
| 2 | 365 | 393 | 6.2 |
| 3 | 1314 | 1345 | 10.6 |
| 4 | 1539 | 1551 | 12.1 |
| 5 | 1649 | 1663 | 13.5 |
| 6 | 1788 | 1798 | 15.3 |
| 7 | 1942 | 1949 | 16.6 |
| 8 | 2034 | 2052 | 17.7 |
| 9 | 2177 | 2196 | 19.2 |
| 10 | 2286 | 2333 | 22.2 |
| 11 | 2886 | 2929 | 26.5 |
| 12 | 3261 | 3270 | 27.4 |
| 13 | 3360 | 3368 | 28.5 |
| 14 | 3492 | 3501 | 30.5 |
| 15 | 3745 | 3761 | 33.1 |
| 16 | 3951 | 3960 | 34.3 |
| 17 | 4026 | 4034 | 35.4 |
| 18 | 4167 | 4174 | 36.3 |
| 19 | 3645 | 3661 | 36.9 |
| 20 | 3851 | 3860 | 38.1 |
| 21 | 3926 | 3934 | 39.3 |
| 22 | 4067 | 4074 | 41.3 |
| 23 | 4160 | 4168 | 42.9 |
| 24 | 4292 | 4300 | 45.5 |

# 5.0 TEMPEST INPUT MODELS AND ASSUMPTIONS

This chapter begins with a discussion of the considerations taken into account in developing the TEMPEST input models for this study. Next, each of the models developed to predict and compare results is described. Last, the most important assumptions underlying each model's development are discussed.

## 5.1 MODEL DEVELOPMENT CONSIDERATIONS

Five different glass pouring and filling scenarios were simulated: the 12LI and 12HI canisters, the 13-in. and 24-in.-diameter canisters, and the 12-in.-diameter TTS canister. The 12LI and 12HI models were batch-pour cases, while the 13-in.-diameter canister model simulated a continuous pour. The 24-in.-diameter canister was filled in 33 separate batch pours, which were modeled with the continuous-fill version of TEMPEST (Version M). The TTS canister was subjected to a batch-pour scenario.

The number of cells used to model a given canister depends primarily on whether the canister is to be modeled with the batch-fill model (Version L4X) or the continuous-fill model (Version M) of TEMPEST. The continuous-fill model required uniform axial cells and allowed only 10 cells in the radial direction for the glass-filling region. The batch-fill model had no restrictions on the number and dimensions of the radial and axial cells in the glass.

The production of time history plots with TEMPEST is accomplished by dumping the temperature of user-selected cells to an auxiliary post-processing file each time step. These user-selected cells are called monitor cells, and up to 16 cells can be selected on any run.

When the TEMPEST input is designed, care must be taken to arrange for the center of a cell to come as close as possible to the actual spatial location of the thermocouples in each cell. In addition, cells representing the response of thermocouples should not be too large, because the temperature calculated by TEMPEST is actually the average temperature in a cell. For stability reasons, variations in axial and radial cell dimensioning should increase and decrease as gradually as possible.

The meshes designed to handle the batch-filling problems have another restriction. A batch pour is simulated by instantaneously converting cells that used to be air into cells that are glass. This is done by stopping the code, changing the input, and doing a restart. Because the batch-pour version of the code (L4X) does not allow for partially filled cells, certain axial dimensions must be chosen to precisely match the amount of glass in an individual pour.

Although TEMPEST is a three-dimensional code, all of the models described in this chapter were for two-dimensional meshes. Except for the TTS case, any variation in the circumferential direction was insufficient to justify a full three-dimensional simulation. However, a major objective of the TTS simulation was to see how accurately a two-dimensional model would work.

## 5.2 INPUT MODELS

The TEMPEST models used to simulate the scenarios are described in this section.

### 5.2.1 12LI and 12HI Canisters

Figures 5.1 and 5.2 show the TEMPEST noding used for the 12LI canisters. Modeling was performed using the batch-fill Version L4X of TEMPEST. In this version, the glass is modeled as a solid, without accounting for any glass convection during its molten state. Figure 5.1 shows the TEMPEST mesh with glass height after the first batch pour. Cells above the glass surface are noncomputational. Figure 5.2 shows the same mesh with the glass height after the second and the third (final) pour. Again, cells above the glass surface are noncomputational.

The 16 monitor cell locations for each modeled canister were determined on the basis of the number of axial levels and locations of thermocouples in each experimentally modeled canister. The batch-fill modeled canisters each had eight thermocouple levels; locations of the 16 monitor cells used for canisters 12HI and 12LI are listed in Table 5.1.

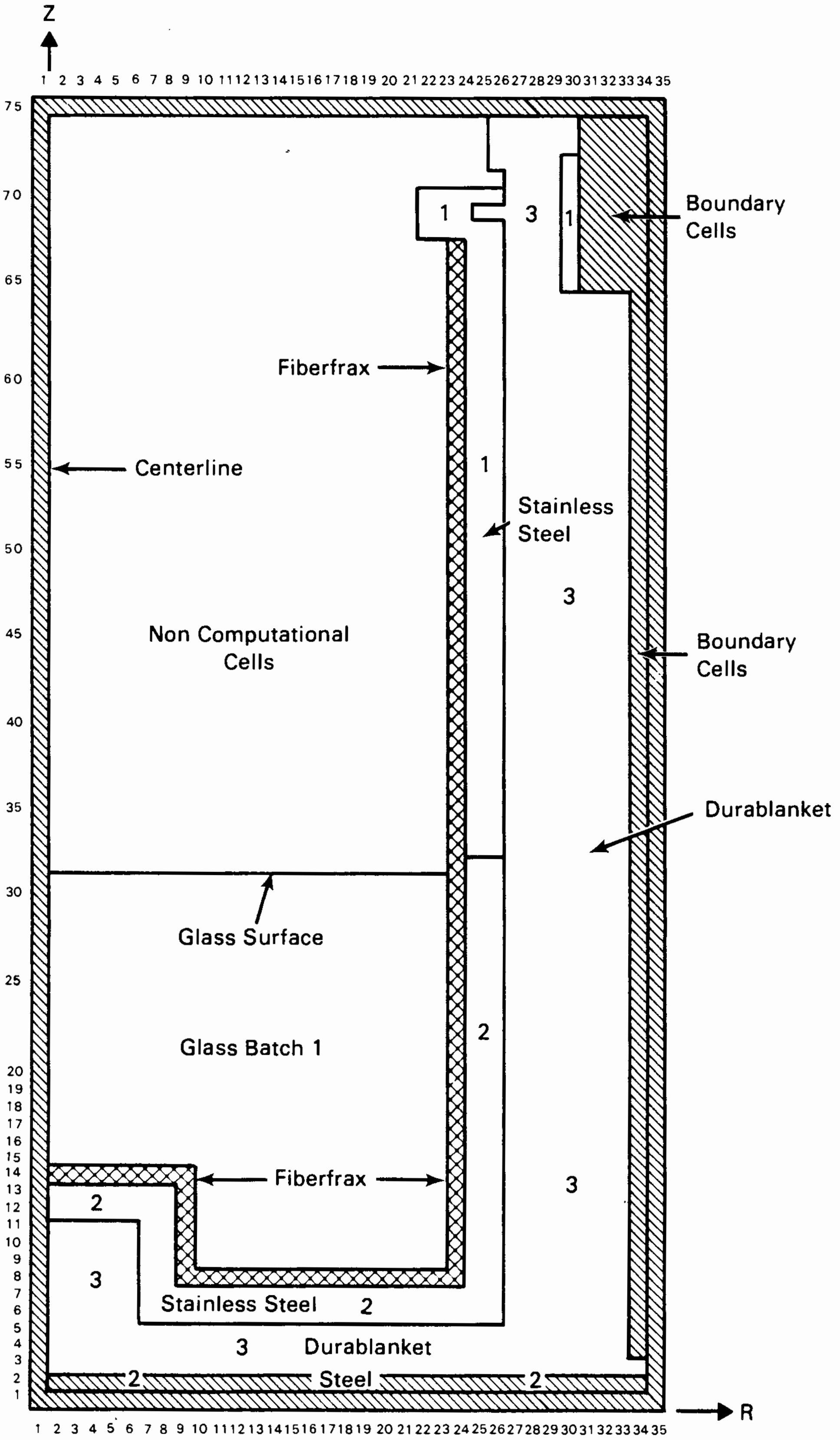


FIGURE 5.1. TEMPEST Mesh for Canisters 12LI and 12HI After First Glass Batch Pour

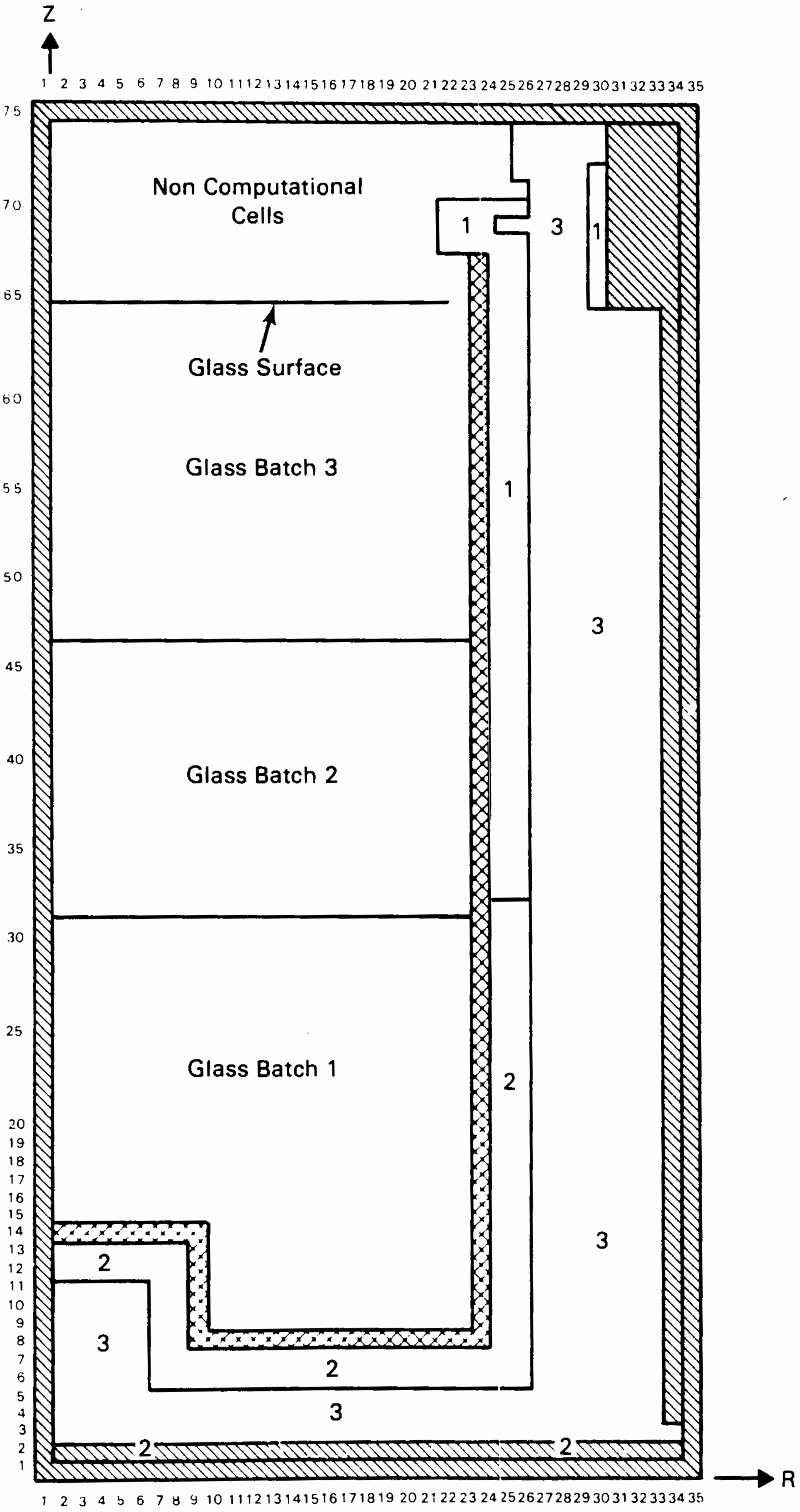


FIGURE 5.2. TEMPEST Mesh for Canisters 12LI and 12HI After Final (Third) Glass Batch Pour

TABLE 5.1. Monitor Cell Locations for Canisters 12LI and 12HI

| Axial Level | Radial Positions Monitored (in from the inside wall) | | |
|---|---|---|---|
| 1 | 4.25 in. | | 1 in. |
| 2 | 4.25 in. | | 1 in. |
| 3 | 4.25 in. | | 1 in. |
| 4 | 4.25 in. | 2 in. | 1 in. |
| 5 | 4.25 in. | | 1 in. |
| 6 | 4.25 in. | | 1 in. |
| 7 | 4.25 in. | | |
| 8 | 4.25 in. | | 1 in. |

### 5.2.2 13-in.-Diameter Canister

Figure 5.3 shows the TEMPEST noding for the 13-in.-diameter canister. Note that this model uses the continuous-fill Version M of TEMPEST. This version accounts for convecting glass (with the inviscid approximation described earlier) but does not allow convection of air above the glass surface.

For the 13-in.-diameter canister, there were six axial thermocouples levels, each containing four internal thermocouples (4.25 in., 2 in., 1 in. and 1/8 in. from the canister wall). Because of the constant axial dimension required for the continuous-fill model, it was not possible to match the monitor cells with the location of the thermocouples as closely as for the batch-fill case, and no attempt was made to vary the cell dimension around the area of the thermocouple to minimize the volume of a monitor cell. Locations of the 16 monitor cells for the 13-in.-diameter canister are listed in Table 5.2.

### 5.2.3 24-in.-Diameter Canister

The noding for the 24-in.-diameter canister is very similar to the mesh for the 13-in. canister shown in Figure 5.3. However, in this model, there were 10 axial levels containing four thermocouples per level (10 in., 5 in., 2.5 in. and 0.25 in. from the canister wall). This canister, although filled with 33 batch pours, was also modeled with the continuous-pour Version M of

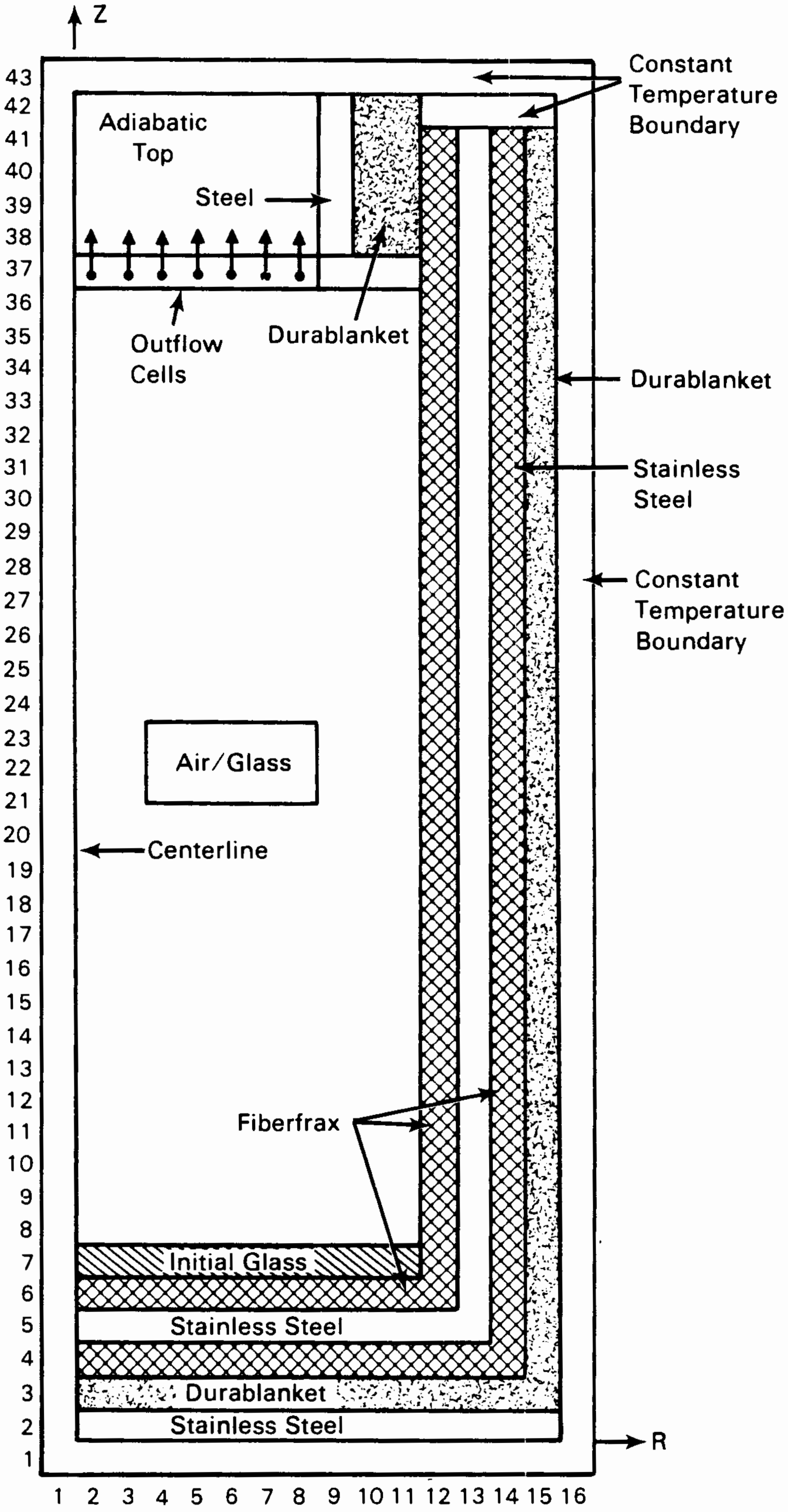


FIGURE 5.3. TEMPEST Mesh for Continuous-Fill Modeling of the 13-in-Diameter Canister

TABLE 5.2. Monitor Cell Locations for the 13-in.-Diameter Canister

| Axial Level | Radial Positions Monitored (in from the inside wall) | | |
|---|---|---|---|
| 1 | 4.25 in. | 1 in. | Glass interface between Fiberfrax liner and glass |
| 2 | 4.25 in. | | Glass interface |
| 3 | 4.25 in. | 1 in. | Glass interface |
| 4 | 4.25 in. | 1 in. | Glass interface |
| 5 | 4.25 in. | | Glass interface |
| 6 | 4.25 in. | 1 in. | Glass interface |

TEMPEST. In this model the cells above the glass surface are noncomputational. Locations of the 16 monitor cells for the 24-in.-diameter canister are listed in Table 5.3.

TABLE 5.3. Monitor Cell Locations for the 24-in.-Diameter Canisters

| Axial Level | Radial Positions Monitored (in from the inside wall) | |
|---|---|---|
| 1 | 10 in. | Glass interface between Fiberfrax liner and glass |
| 2 | 10 in. | |
| 3 | 10 in. | Glass interface |
| 4 | 10 in. | |
| 5 | 10 in. | Glass interface |
| 6 | 10 in. | Glass interface |
| 7 | 10 in. | |
| 8 | 10 in. | Glass interface |
| 9 | 10 in. | |
| 10 | 10 in. | Glass interface |

## 5.3 UNDERLYING ASSUMPTIONS

TEMPEST is a flexible computer code, allowing the user to simulate many different situations. However, as with all computer codes, any simulation of an actual engineering system does involve some approximations. The most important assumptions used in developing the TEMPEST models for canister filling and cooling are discussed in this section.

### 5.3.1 Geometry

The major assumptions related to problem geometry are described in the following paragraphs.

When the two 12-in.- and the one 13-in.-diameter canisters were under the melter, they were positioned beneath a 4-ft spoolpiece that was insulated with 3 in. of Durablanket insulation to prevent any significant heat transfer. However, the spoolpiece had no thermocouple monitoring. To model these three canisters, it was decided to ignore the spoolpiece and treat the area above the modeled canister as adiabatic. The spoolpiece was omitted from the canister models for two reasons: 1) the significant increase in total cells required to accurately handle the spoolpiece and canister, and 2) the uncertainty with respect to actual spoolpiece temperatures.

While under the melter, the canisters sat on a large metal canister stand that was spaced to allow convective and radiative heat transfer underneath. To model this area, it was assumed that while the canister is under a melter it would rest upon a 1/4-in. piece of metal. The metal, having a free convection and radiation coefficient, would then act as a thermal connection to the environment. All three canisters would also be separated from the metal piece by a 1/2-in. Durablanket insulation layer. However, the 12-in.-diameter canisters had a lip and a large inverted bump on the bottom (see Figure 4.1). Likewise, the feet on the 13-in.-diameter canister held the canister above the stand. Rather than simulate a dead air space, the gap was assumed to be filled with Durablanket insulation. Any radiation in this gap was ignored.

When these canisters were moved out from under the melter, they were placed upon a layer of Durablanket insulation on the floor. A canister lid and/or layer of Durablanket insulation was then placed on the canister top. To model this change in node dimensions, a restart of the canister simulation at its change-out time was required. In the restart condition, a canister lid and a layer of Durablanket insulation at room temperature was placed on top of the canister, and axial heat transfer was allowed above the canister's top surface. On the bottom of the model, Durablanket insulation at room temperature was placed between the canister and the 1/4-in. metal shell, and direct conductive heat transfer from the canister to the floor at ambient temperature was used (no convective or radiative heat transfer). On the two 12-in.-diameter canisters, room temperature insulation was also used to fill the canister gap. These changes were assumed to occur instantaneously upon change-out, without changing the previously generated temperatures in the canister. The 12-in.-diameter canisters also had a connecting metal band (holding the canister to the spoolpiece) removed during the change-out restart.

The batch-fill model assumed that air cells (above the current glass surface) change instantaneously into glass cells with the next batch pour. Because the model did not contain a method for changing cell and material types during the computer simulation, a restart of the transient simulation was required for modeling each batch. The restart input file contains the information about changed cell and material types for the section of newly poured glass. In addition, a new initial temperature for the just-added glass volume was required. A disadvantage of the batch-fill model is that it allows only one type of material in each cell (no partially filled cells), and it requires a separate restart for each batch pour. This is a significant inconvenience if the number of batch pours is large.

The continuous-fill model uses a time-dependent boundary table to handle the mass and energy addition of a batch pour or a continuous pour. Theoretically, any arbitrary function can be simulated, including a function that intermittently goes to zero. (Thus, a batch-fill scenario can be simulated.) As a result, Version M of this TEMPEST does not require a restart for every new pour or change in the average fill rate. A restart is required only when a change in environment occurs (e.g., when the canister is moved from under the melter. Filling input for the 13-in.-diameter canister is generated as a series of step-function average fill rates, with glass levels determined by thermocouple response. Filling input for the 24-in.-diameter canister is set up as a series of step functions, alternating between the average fill rate of each batch and no filling.

The 24-in.-diameter canister noding differs from the other canister noding descriptions in that it is not externally insulated and is placed in a 48-in.-diameter turntable simulant shell. The change in emissivity over time on the canister and shell surfaces is uncertain, and the enclosure radiation between the canister and turntable shells cannot be accurately modeled. A time-dependent model was used for the wall temperature of the canisters to fix the heat transfer driving force out of the canister. This time-dependent canister wall temperature was determined by digitizing experimental temperatures taken at ten levels on the 24-in.-diameter. Because of these time-dependent temperatures, no radiation or convection coefficients are required on the canister wall. Adiabatic heat transfer was assumed above the canister, and only conduction heat transfer was assumed out of the bottom of the canister. No restart condition upon canister change-out was required. One additional feature of the 24-in.-diameter canister was the presence of a bowed, inverted bottom. To model the bowed bottom using TEMPEST, an average height of the bow was determined and used as an insulation-filled gap between the canister and turntable shell.

### 5.3.2 Glass Properties

An important part of the comparison between actual temperature profiles and TEMPEST predictions was to determine any significant variation in glass properties as a function of temperature; such variations could affect the TEMPEST predictions. Currently, data on the density, specific heat, and thermal conductivity variations of nuclear waste glass with temperature have been tabulated for SRL-411 glass only. To calculate the thermal properties of another waste glass, properties of a different simulated glass from PSCM-20 were determined at temperatures up to 800°C. A comparison between the PSCM-20 and the old SRL 411 glass properties is shown in Table 5.4.

As shown, certain assumptions were made to extrapolate the measured PSCM-20 values over the full range of the SRL 411 table. The extended density values above 600°C should be quite accurate, as the SRL values have a thermal

TABLE 5.4. Comparison of SRL 411 and PSCM-20 Glass Properties

| Temperature, °C | Density, $kg/m^3$ | | Specific Heat, J/kg-K | | Thermal Conductivity, W/m-K | |
|---|---|---|---|---|---|---|
| | SRL 411 | PSCM-20 | SRL 411 | PSCM-20 | SRL 411 | PSCM-20 |
| 22 | 2610 | 2801 | 920 | 696[a] | 1.05 | 0.813[a] |
| 100 | 2603 | 2794 | 1000 | 757 | 1.11 | 0.858[a] |
| 200 | 2598 | 2785 | 1100 | 892 | 1.16 | 0.921 |
| 300 | 2590 | 2776 | 1180 | 983 | 1.16 | 1.021 |
| 400 | 2575 | 2767 | 1270 | 1060 | 1.22 | 1.095 |
| 500 | 2555 | 2757 | 1910 | 1180 | 1.63 | 1.144 |
| 600 | 2531 | 2718 | 1820 | 1710 | 1.17 | 1.006 |
| 700 | 2501 | 2678[b] | 1520 | 1710[c] | 1.05 | 0.864 |
| 800 | 2458 | 2638[b] | 1500 | 1710[c] | 1.20 | 0.842 |
| 900 | 2405 | 2599[b] | 1610 | 1710[c] | 1.48 | 0.994[a] |
| 1000 | 2352 | 2561[b] | 1730 | 1710[c] | 1.83 | 1.153[a] |
| 1100 | 2321 | 2523[b] | 1880 | 1710[c] | 2.23 | 1.288[a] |
| 1200 | 2310 | 2487[b] | 2020 | 1710[c] | 2.62 | 1.399[a] |

[a] Extrapolated or projected using SRL-411 values.
[b] Assumes the same thermal expansion at all temperature >506°C.
[c] Assumes constant specific heat above 600°C (1710 J/kg-K).

expansion coefficient similar to that of the PSCM-20 glass at these temperatures. The assumed specific heat values are based on a number of curves of glass specific heat versus temperature, which have shown a constant specific heat for glass above 650°C. Although possibly inaccurate, the inaccuracy is offset at these temperatures if the thermal diffusivity values ($k/\rho Cp$) are accurate. All other iterated values are determined by projecting the property slopes from SRL 411 glass over the PSCM-20 glass. This is an area where significant potential for error exists and is a possible cause for TEMPEST deviations from experimental values.

A comparison of the thermal properties shows notable variances, particularly with regard to the thermal conductivity and specific heat properties. However, the effect on thermal diffusivity values is negligible except at temperatures above 500°C where the measured PSCM-20 properties become suspect. Although better property measurements, particularly in the higher temperatures, would be desirable, it can be assumed that the thermal property variations in glasses above 500°C do not significantly affect steady-state heat transfer.

Because the new measured glass properties were from PSCM-20, it was decided to use these measured and assumed values for canister 12HI and 12LI, which were modeled with the batch-fill version of TEMPEST (L4X). With the 13- and 24-in.-diameter canisters from PSCM-21, however, it was felt that PSCM-21 glass would resemble SRL 411 glass more closely than PSCM-20 glass. Therefore, glass properties used in the continuous-fill model consisted mostly of SRL 411 properties. Densities were increased by a factor of 1.034 to account for the increased PSCM-21 glass density at room temperature (2700 kg/m$^3$).

A glass emissivity of 0.8 was measured for the SRL 411 glass. No mention was made of the effect of temperature on glass emissivity. However, it was felt that the 0.8 emissivity would be sufficiently accurate for use with TEMPEST's crude glass surface radiation model. If development proceeds on a better radiation heat transfer model, it would be wise to measure glass emissivity as a function of temperature.

### 5.3.3 Insulation Properties

Temperature-dependent thermal conductivity data for Durablanket and Fiberfrax and emissivity data for Fiberfrax (970-J) were taken from vendor information. However, the temperature dependency of the insulation's specific heat and density is considered proprietary information by the manufacturer. Therefore, a few assumptions had to be made.

For uncompressed insulation, the 8-lb/ft3 bulk density for Durablanket and 10-lb/ft$^3$ bulk density for Fiberfrax are fractions of each material's specific gravity. It was assumed that any thermal expansion of the insulation material would serve only to increase or decrease the fraction of the material's specific gravity without changing bulk volume. This assumption justifies the use of a constant bulk density versus temperature for the two insulations.

The assumptions used to derive a temperature-dependent specific heat for the two insulation materials were iterative and more approximate. Both insulations are composed of ~50% $SiO_2$ and ~50% $Al_2O_3$. Temperature-dependent specific heat tables for the two nonmetallic materials were found, and an average specific heat at each temperature point was determined. The new average specific heat table was used for the two insulation values.

An additional consideration brought about in determining insulation properties is the compression of insulation in the canister by the glass fill. The primary compression is caused by the hydrostatic pressure forces of the glass. However, the insulation compression can be influenced also by thermal expansion differences between the canister wall and the glass. Average calculations of insulation compression on the canister floor and walls, based on determined compression thicknesses, resulted in the average insulation thickness given in Table 5.5 (wall thicknesses are averages). The insulation dimensions shown in Table 5.5 were used in the TEMPEST simulations.

The insulation manufacturer had no information on the effect of compression on insulation thermal conductivity. For this study, no change was assumed. For new bulk density values, the percentage of compression was added to 1.00 and multiplied by the uncompressed bulk density. Specific heat was assumed not to change with compression since specific heat is dependent on material weight alone.

Vendor information provided a constant value of 0.9 for the Durablanket insulation emissivity. Although this may rise with increasing temperature, it was assumed constant for this study because the highest temperature seen by the radiating Durablanket surface is relatively low.

### 5.3.4 Stainless Steel Properties

Thermal properties of 304L stainless steel versus temperature are well documented (JANAF 1986). The primary concern with stainless steel is the influence of temperature on oxidation rate and the influence of oxidation rate on canister shell emissivity. More information is needed in this area. Stainless steel emissivity data are necessary for only the 24-in.-diameter canister simulation, because the radiation heat transfer from all other canister surfaces was controlled by the emissivity of the Durablanket.

TABLE 5.5. Canister Insulation Thickness

A. 12- and 13-in.-Diameter Canister Insulation Thicknesses

| Dimension | Uncompressed | Compressed |
|---|---|---|
| Bottom bump (12 in.) | 0.125-in. | 0.1045-in. |
| Bump wall (12 in.) | 0.125-in. | 0.1035-in. |
| Canister bottom | 0.125-in. | 0.1035-in. |
| Canister wall | 0.125-in. | 0.106-in. for 12HI<br>0.121-in. for 12LI |

B. 24-in.-Diameter Canister Insulation Thicknesses

| Dimension | Uncompressed | Compressed |
|---|---|---|
| Canister bottom | 0.5-in. | 0.349-in. |
| Canister walls | 0.25-in. | 0.212-in. |

### 5.3.5 Initial Glass Temperatures

Initial glass temperatures for the continuous-fill canisters were estimated at 1075°C. This estimate was based on observed pouring temperatures for the 13-in.-diameter canister. No effort was made to estimate glass temperatures more accurately because there were not enough temperatures or levels being recorded in each can.

Temperatures for the batch-fill modeled canisters were determined as follows:

- If the experimental temperatures were recorded every 2 minutes, the initial glass temperature used for a particular pour was an average of the hottest temperature (usually centerline) recorded at each axial level covered during the pour.
- If the experimental temperatures were recorded every 6 minutes, the initial glass temperature used for a particular pour was the hottest temperature recorded during a pour.

Table 5.6 is a summary of the initial glass temperatures used for modeling canisters 12LI and 12HI.

TABLE 5.6. Initial Glass Temperatures Used for Canistera 12LI and 12HI During Batch-Fill Modeling

| Pour Number | 12LI | 12HI |
|---|---|---|
| 1 | 1056°C | 1115°C |
| 2 | 1050°C | 1028°C |
| 3 | 1060°C | 919.9°C |

The pour temperatures used are within 50°C of 1075°C, with the exception of the last pour temperature of canister 12HI. This low temperature was caused by the loss of a face heater in the glass overflow section of the melter during pouring. It was determined as the average of the hottest recorded temperatures for three axial levels in the glass pour.

# 6.0 COMPARISON OF TEMPEST PREDICTIONS WITH EXPERIMENTAL OBSERVATIONS

In this chapter, temperature predictions generated with the TEMPEST input models are compared to canister temperature measurements obtained during actual filling of canisters by the PSCM. The TEMPEST-predicted heat transfer behavior of the turntable-filled canister is also compared with actual observations.

## 6.1 CANISTER INTERNAL TEMPERATURES

Canisters 12LI, 12HI, and the 24-in.-diameter canister were filled with batch pours, but the 13-in.-diameter canister was filled in a continuous pour. Canisters 12LI and 12HI were filled in three pours; the 24-in.-diameter canister was filled in 33 pours. Canisters 12HI and 12LI were modeled using the batch-fill version of TEMPEST (L4X); the 13-in.- and 24-in.-diameter canisters were modeled using the continuous-fill version of TEMPEST (M). The continuous-fill version was selected for the 24-in.-diameter canister because modeling the 33 batch pours using the batch-fill version would have required too much staff time.

### 6.1.1 TEMPEST Batch-Fill Model

Two radiation models were used to compute the radiation heat transfer from the glass surface to the Fiberfrax liner in the canister. The first used the temperature of the Fiberfrax in contact with the glass surface as the sink temperature.[a] Because the actual sink temperatures are less than the computed sink temperature, the rate of heat transfer from the glass surface is underpredicted. Because this would result in a high-temperature prediction, this model is referred to here as the high-temperature model. Using the second radiation model, the temperatures of the Fiberfrax in contact with the glass and the Fiberfrax at the top of the canister were averaged for use as the sink temperature. This estimate of sink temperature is probably lower than the mean sink temperature; therefore, the radiant flux for this model is probably

[a] Average sink temperature refers to the temperature that would result in the same radiation to the glass surface from the portion of the canister and melter above the glass as which existed during the pour.

overpredicted. Consequently, the predicted glass temperatures should be low. Hence, this model is referred to as the low-temperature model.

Canister 12HI

Canister 12HI was insulated externally with 3 in. of Durablanket insulation and was filled with an "interrupted" three-batch pour. After one-third of the canister was filled in one batch, the canister was removed from under the PSCM for 1 day. The first pour was 13.1 in. deep. The canister was then placed back under the PSCM, where the second and third pours were made 29.5 h and 33.95 h after the start of the first pour.

Comparisons between the TEMPEST predictions using the high-temperature radiation model and the observed time/temperature histories are given for the eight axial thermocouple levels in canister 12HI in Figures 6.1 through 6.7. Because this canister was heavily insulated, the radial temperature differences are small during most of the cooldown phase.

The onset of the heating wave from the second pour can be seen in Figures 6.1 through 6.3. Note how the wave reaches the highest axial thermocouple level covered by the first pour very close to the time when the pour started (Figure 6.3), but does not cause a noticeable change in the downward slope of either the predicted or observed temperatures for the other two axial thermocouple levels (Figures 6.1 and 6.2) until 30 and 31 h (0.5 and 1.5 h after the pour). As expected, the temperature rise caused by heating from the second pour decreases with distance between the axial thermocouple level and the top of the first pour.

The TEMPEST predictions shown in Figures 6.1 through 6.3 have the following characteristics:

- They are in reasonable agreement with experimental results.
- The predicted cooldown for the first 8 to 10 h is too slow.[a]
- Agreement of TEMPEST predictions with experimental results improves during the remainder of the 29.5 h prior to the second fill.

[a] As will be discussed later, this is not true for the top batch surfaces.

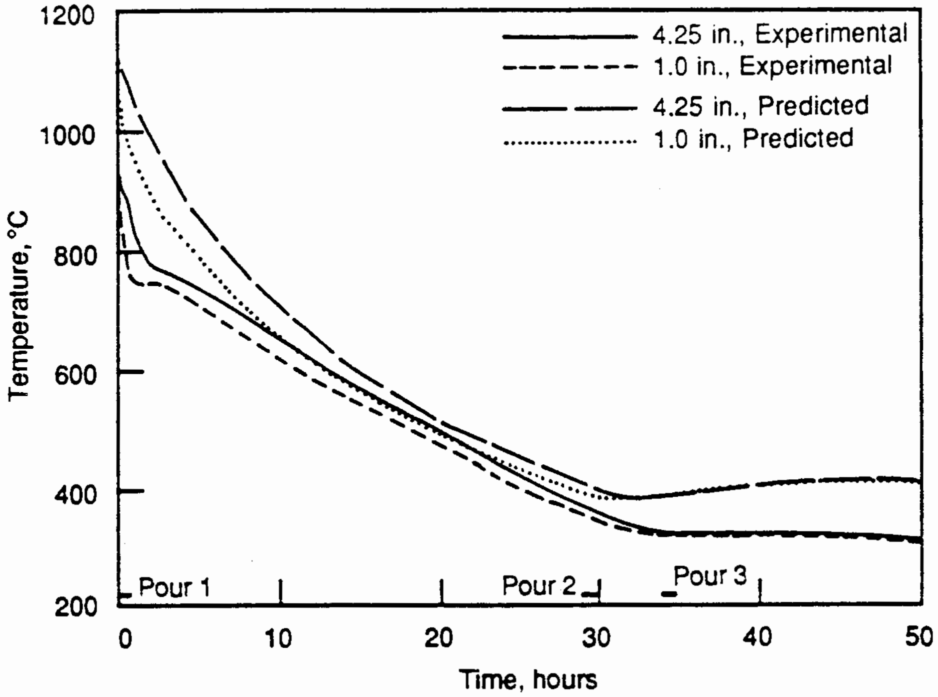


FIGURE 6.1. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 3.75 in. from Canister Bottom

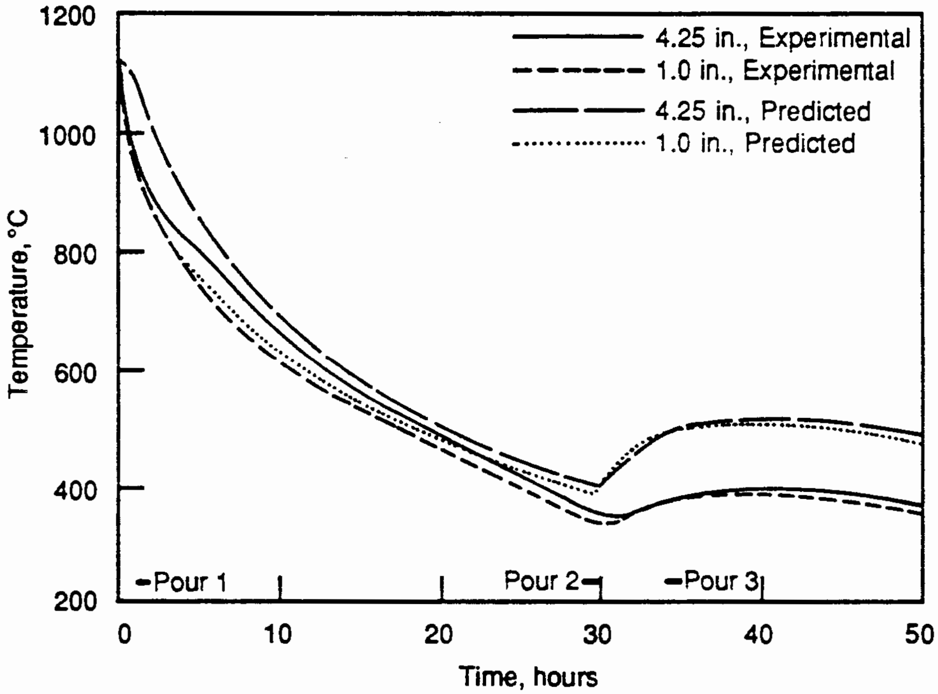


FIGURE 6.2. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 9.5 in. from Canister Bottom

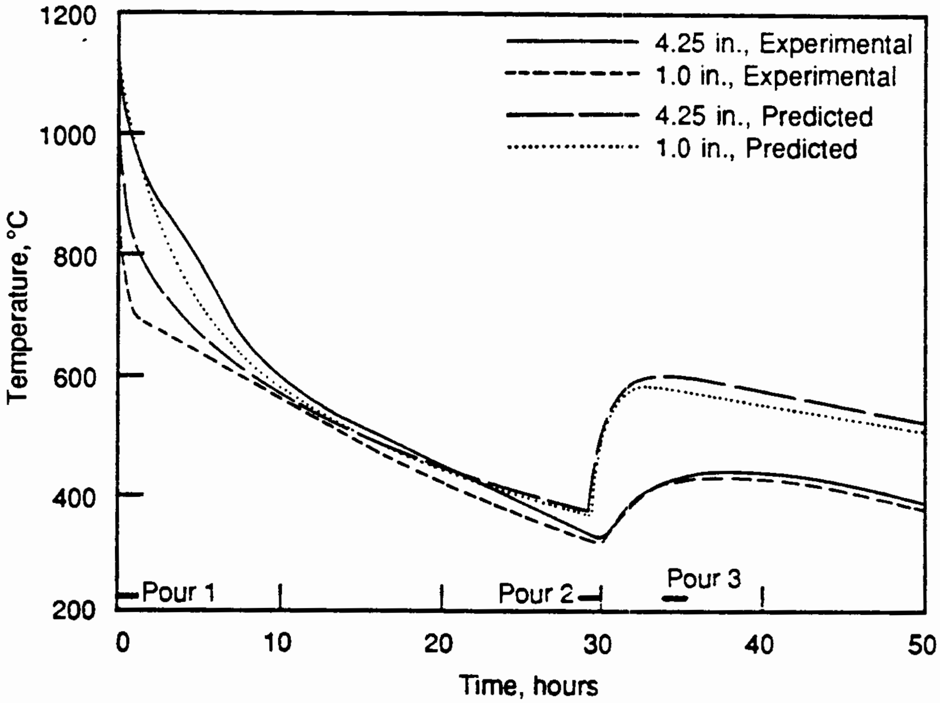


FIGURE 6.3. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 12.125 in. from Canister Bottom

- TEMPEST predicts a considerably greater increase in temperature by reheating from the second pour at the second and third thermocouple levels (Figures 6.2 and 6.3) and a somewhat greater reheat at the first thermocouple level than that observed (Figure 6.1).

Figures 6.4 and 6.5 show the results for the two axial thermocouple levels (20 in. and 24.875 in.) covered during the second pour. The top of the second pour was at 27 in., so these levels were 7.0 and 2.25 in. below the surface of the glass at the completion of the pour. The first of these levels (Figure 6.4) again showed that TEMPEST underpredicts the rate at which the temperature falls immediately after the pour ends. The near parallel behavior between the predicted and experimental results after 35 h makes it appear that almost perfect agreement would be obtained if TEMPEST were to predict the initial cooldown rapidly enough. This is not true, however, because the TEMPEST predictions should show a steeper slope because the predicted temperatures and, hence, heat transfer driving forces, are higher. Although it underpredicts the rate of cooldown just after the pour, TEMPEST appears to accurately predict the

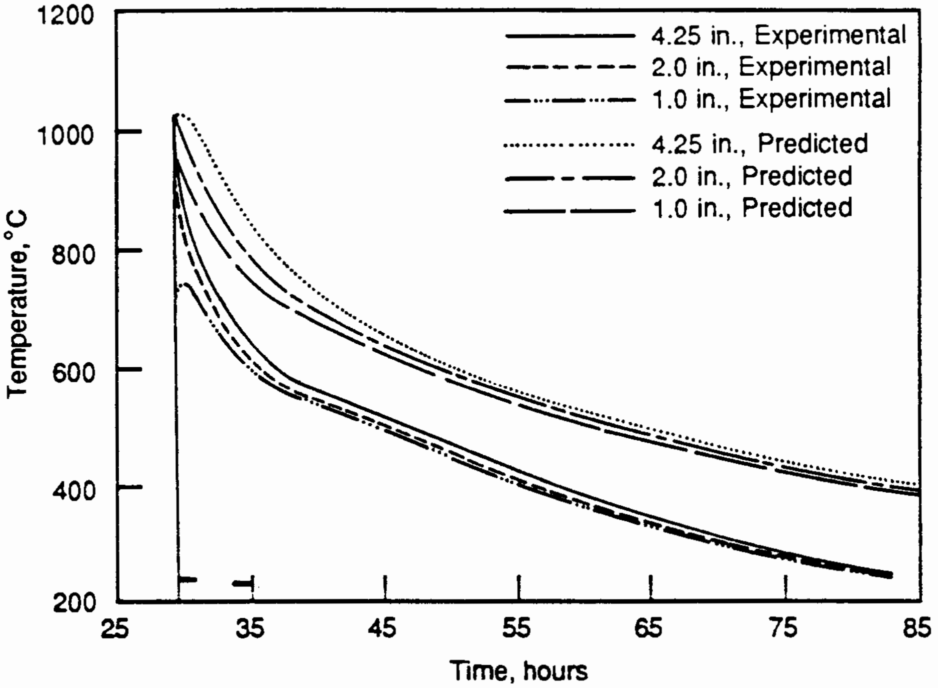


FIGURE 6.4. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 20.0 in. from Canister Bottom

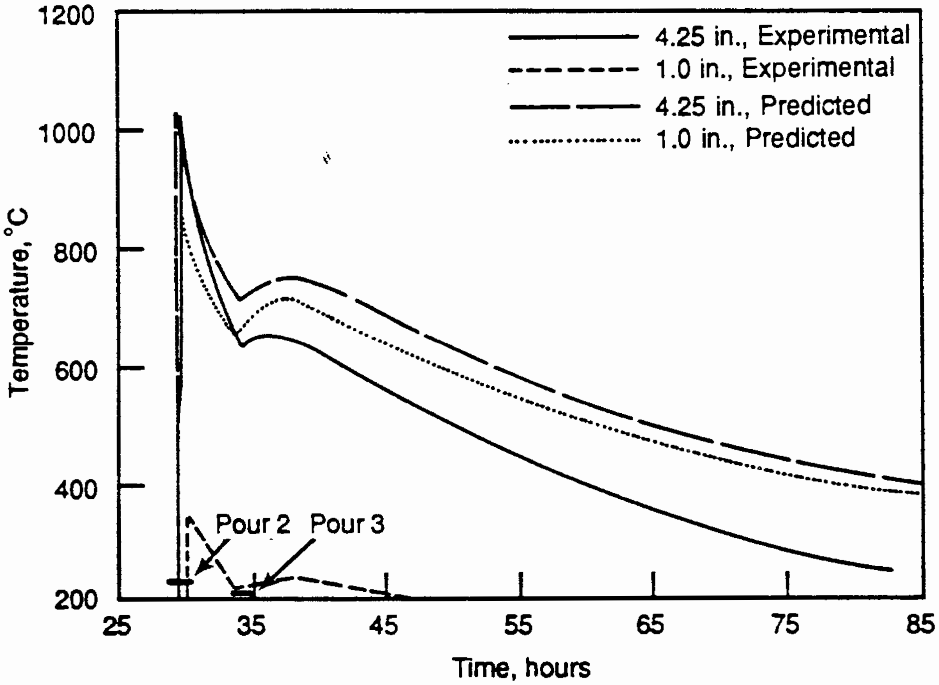


FIGURE 6.5. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 24.875 in. from Canister Bottom

timing and the magnitude of reheat in Figure 6.5 for the 4-1/4-in. position. Thermocouple malfunction at the 1-in. thermocouple located at the 25-in. level caused the very low reading shown in Figure 6.5.

Figures 6.6 and 6.7 show the results for the three axial thermocouple levels (32.75 in., 35.875 in., and 37.625 in.) covered during the final pour. The results for the 37.6-in. thermocouple level are similar to those found during the second pour. Compared to experimental profiles, TEMPEST predicts a much slower cooldown rate early in the transient and a slightly slower cooling rate in the latter stages (evidence by the lower time-temperature slope although the predicted temperature is higher). The final glass level was estimated at 38.5 in. Note that the 4.25-in. thermocouple on the 37.6-in. level was cooler than the 4.25-in. thermocouple on the level below (Figure 6.7); this is due to the heat transfer from the top of the glass.

For the two thermocouples at the 37.6-in. level, TEMPEST predicts an anomaly in the time/temperature history at about 43 h. At this time, the negative slope of the temperature curves increase suddenly, then returns to a more typical value. This is probably due to an inaccurate measurement of the time at which canister 12HI was removed from under the melter.

Comparisons of the observed and predicted glass temperatures for the eight 4.25 in. thermocouples are illustrated in Figures 6.8 through 6.15. The observed glass temperatures were compared with the predicted glass temperatures from both the high and low radiation heat transfer models. As mentioned above, use of these two models would be expected to bound the radiant heat transfer from the surface. The low-temperature model does indeed predict lower glass temperatures. However, the temperatures predicted by the two models differ by less than 55°C over the entire cooling period, except

- during the cooldown from the first pour, at axial thermocouple levels 2 (9.5 in.) and 3 (12.1 in.) (See Figures 6.9 and 6.10)
- at level 5 (24.9 in.), for a short time before and after the final pour (see Figure 6.12) - Although the final pour caused a heat-up peak, the high and low radiation model predictions increased until the final pour. Thereafter, the differences decreased rather

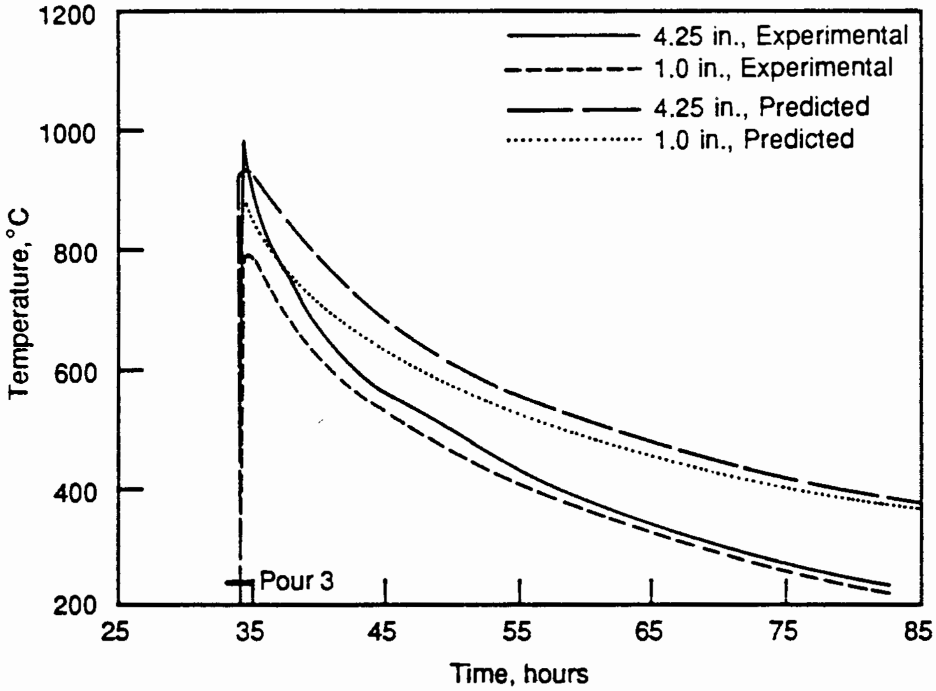


FIGURE 6.6. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 32.75 in. from Canister Bottom

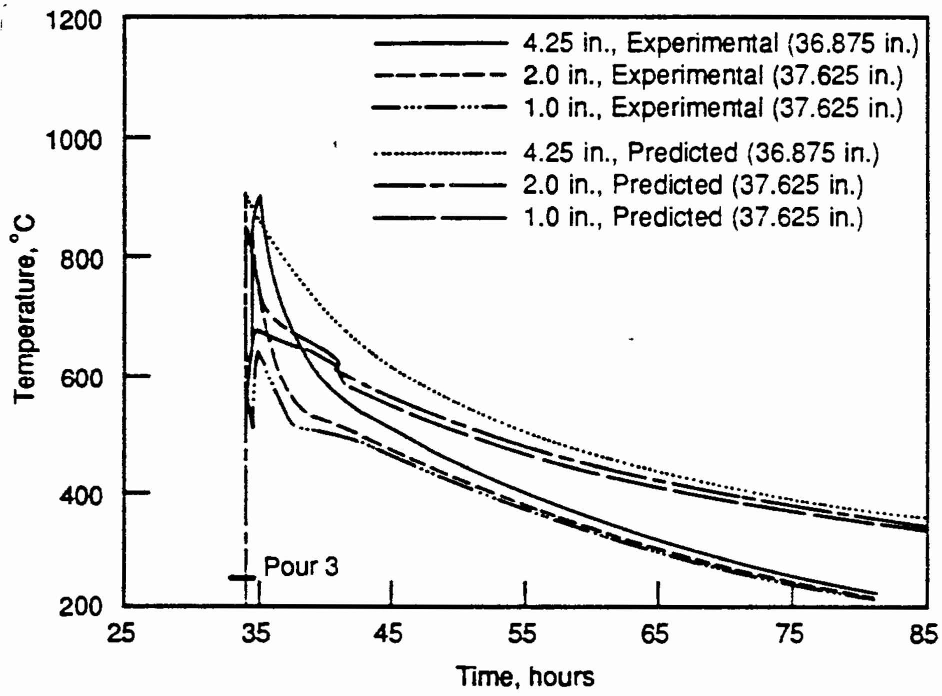


FIGURE 6.7. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Levels 37.625 and 35.875 in. from Canister Bottom

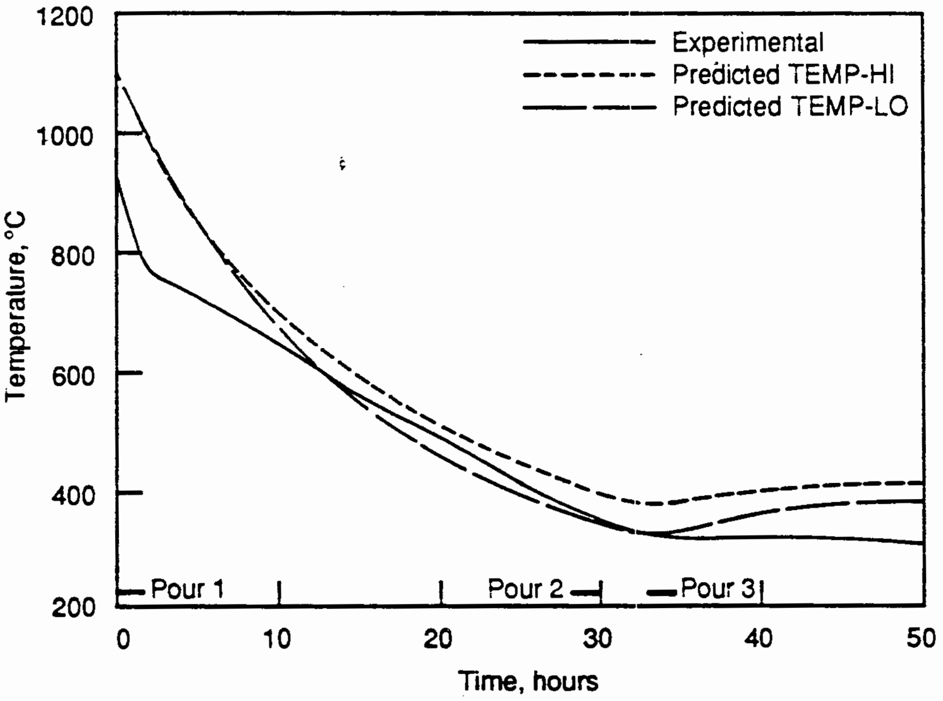


FIGURE 6.8. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 3.75 in. from Canister Bottom

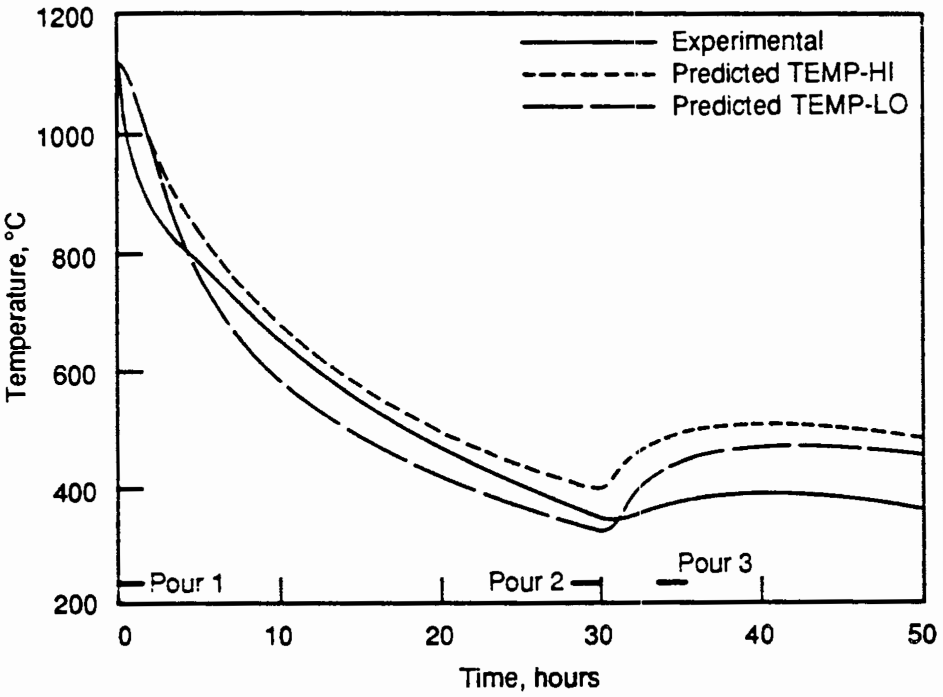


FIGURE 6.9. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 9.5 in. from Canister Bottom

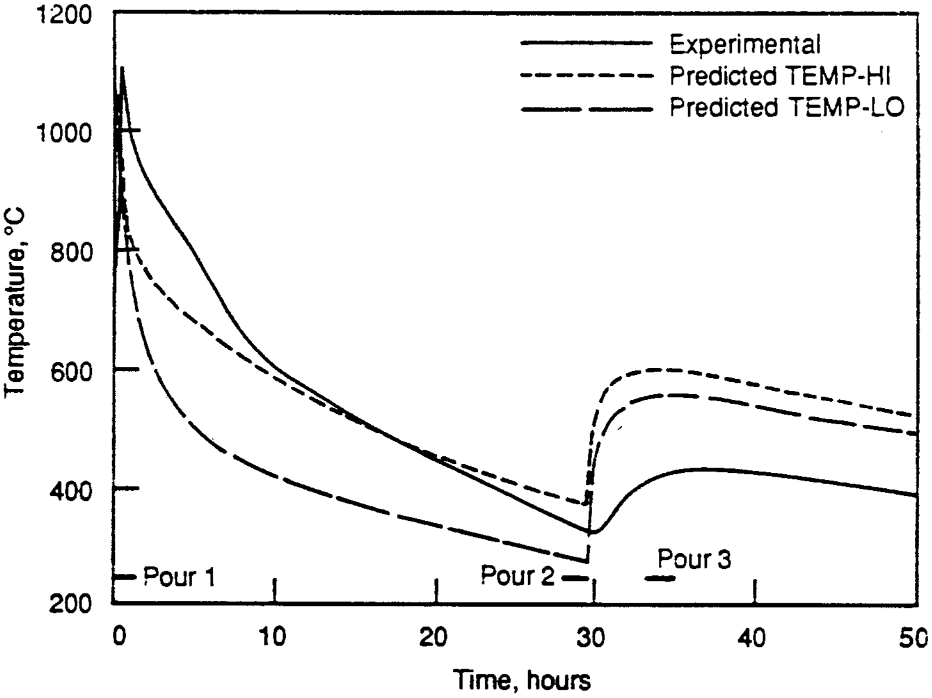


FIGURE 6.10. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 12.125 in. from Canister Bottom

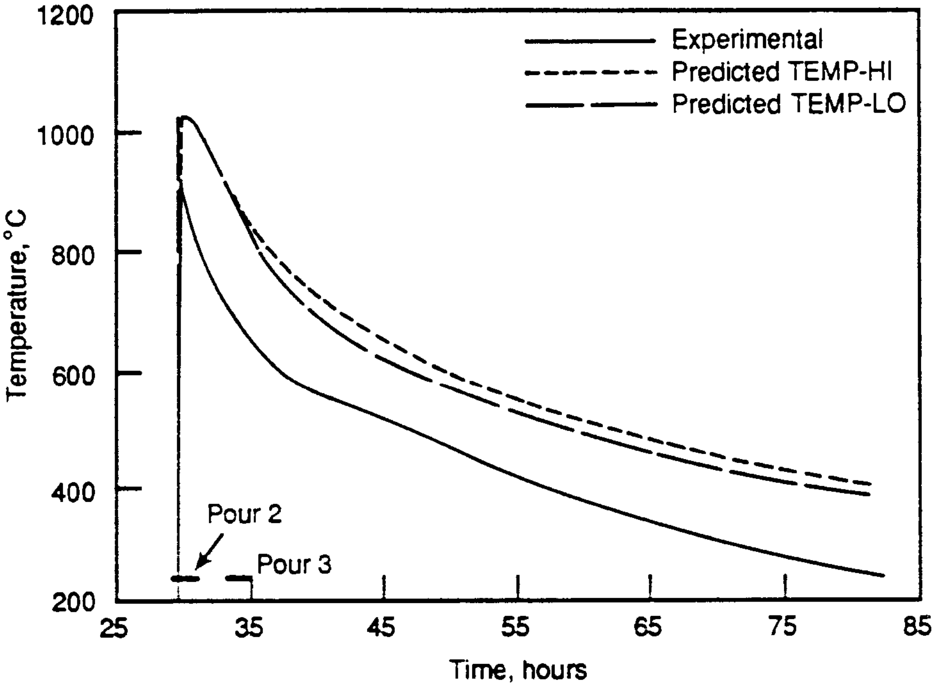


FIGURE 6.11. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 20.0 in. from Canister Bottom (Second Pour)

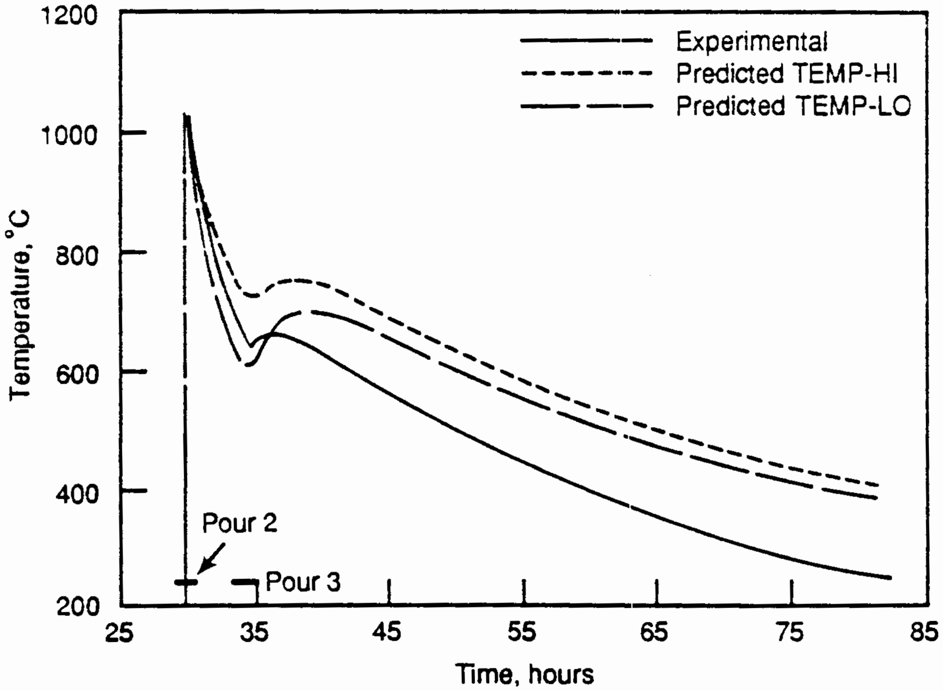


FIGURE 6.12. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 24.875 in. from Canister Bottom

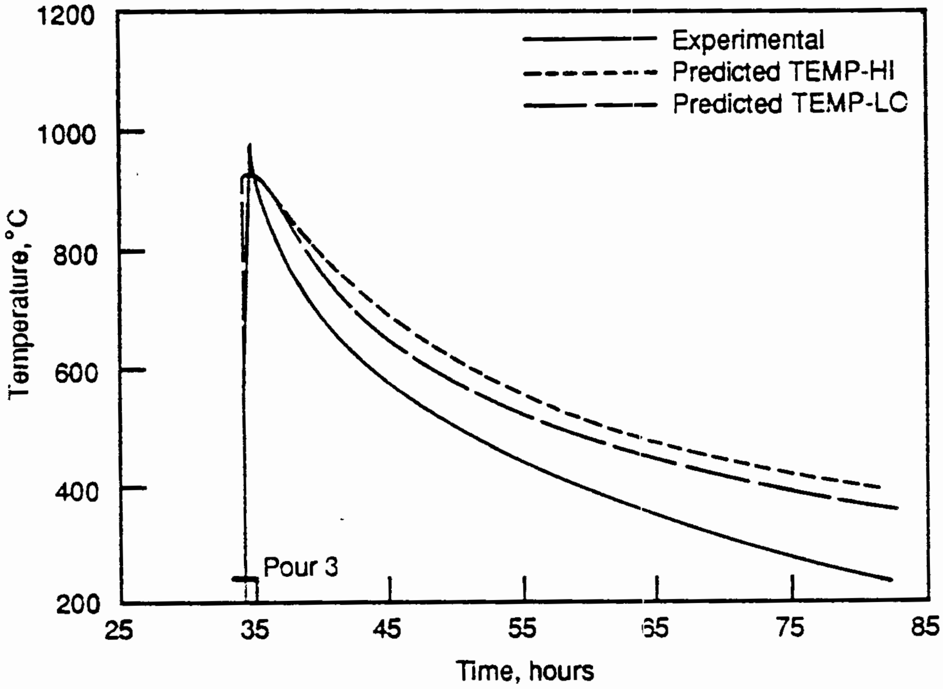


FIGURE 6.13. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 32.75 in. from Canister Bottom

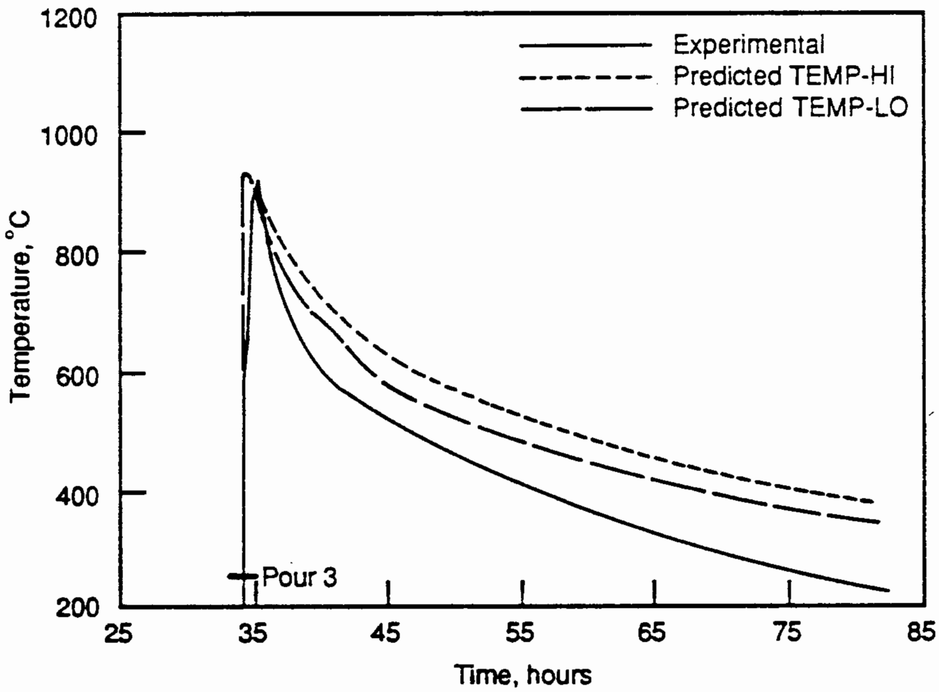


FIGURE 6.14. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 35.875 in. from Canister Bottom

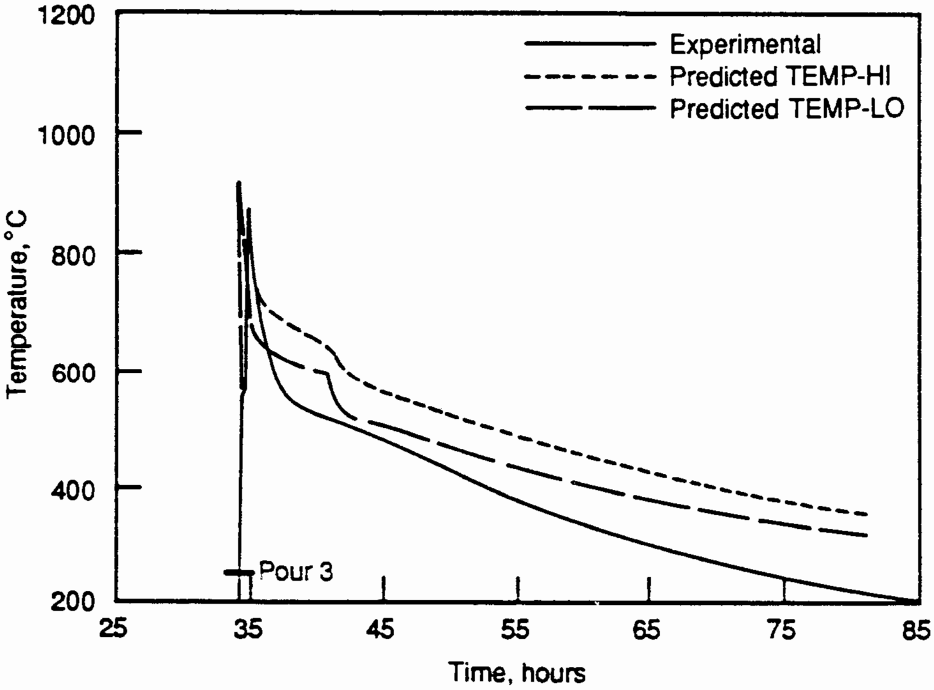


FIGURE 6.15. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12HI at Axial Level 37.625 in. from Canister Bottom

rapidly. This suggests that an error in glass height measurement might explain the large difference in the predictions of the high and low radiation models.

Note that level 3 is 1 in. from the top of pour 1 and level 5 is only 2.2 in. from the top of pour 2. In each case, a greater than 50°C difference occurs between the two models. Because the levels of the first and second pours were known to within an accuracy of only 1 in., it is possible that these large differences between the two radiation models were partially caused by an error in assumed glass level for TEMPEST. For example, if level 3 was only 0.5 in. from the glass surface, the effects of the heat loss from the surface by radiation (as opposed to heat loss through the walls) would be significantly greater than if 1 in. of glass covered the thermocouple. Thus, errors in glass height could well be the explanation for these large differences in results obtained from the two radiation models.

However, for the most part, the differences between the two radiation models were small enough that they did not change the general picture. For canister 12HI, TEMPEST underpredicts the rate of cooldown immediately after a pour and, to a lesser extent, it underpredicts the rate of cooldown thereafter.

Canister 12LI

Canister 12LI was covered with 1/4 in. of Durablanket insulation. This canister was filled in three pours; the second and third batches of glass were poured 5.1 to 6.0 h and 11.9 to 12.7 h, after initiation of the first pour. Glass interfaces between the three pours were at 12.9 in., 25.7 in., and 39.5 in. Because the canister has less insulation, the radial temperature differences were larger and the cooling rates greater than those of canister 12HI.

Comparisons between the temperatures predicted by TEMPEST using the high-temperature radiation model and the temperatures observed at the eight axial thermocouple levels are presented in Figures 6.16 through 6.22. Unlike canister 12HI, the onset of heating from the second pour could not be detected at the first thermocouple level (3.75 in. as shown in Figure 6.16), although the amount of glass added during the first pour was nearly the same for both

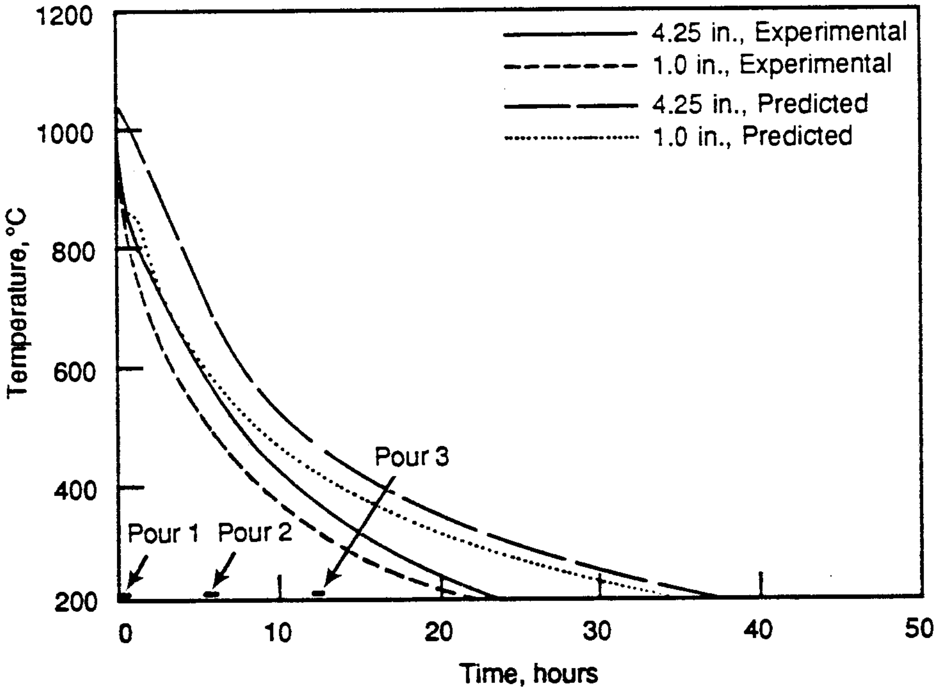


FIGURE 6.16. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 3.75 in. from Canister Bottom

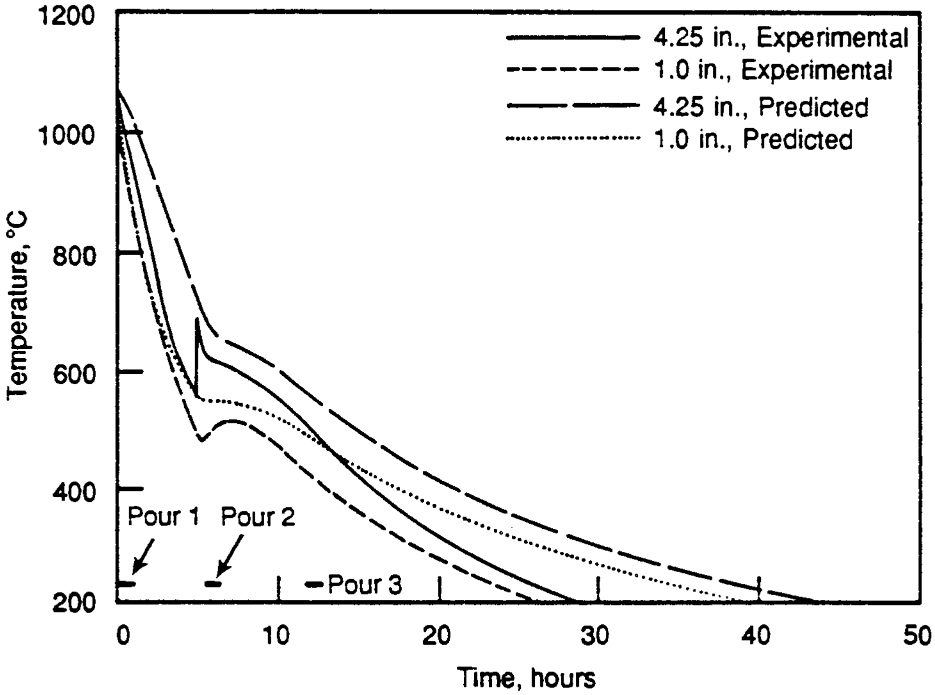


FIGURE 6.17. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 9.50 in. from Canister Bottom

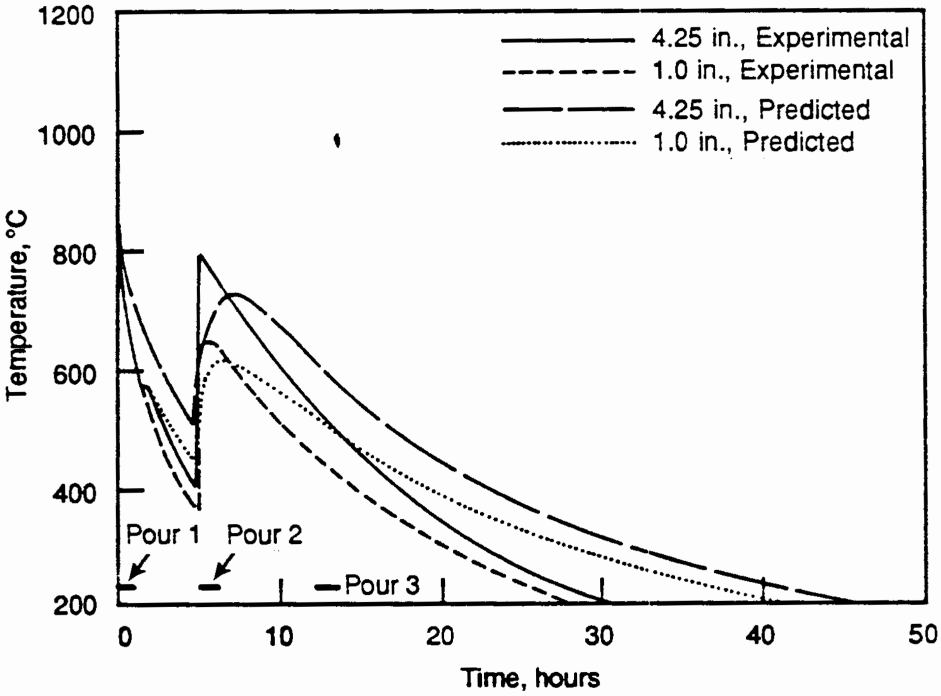


FIGURE 6.18. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 12.175 in. from Canister Bottom

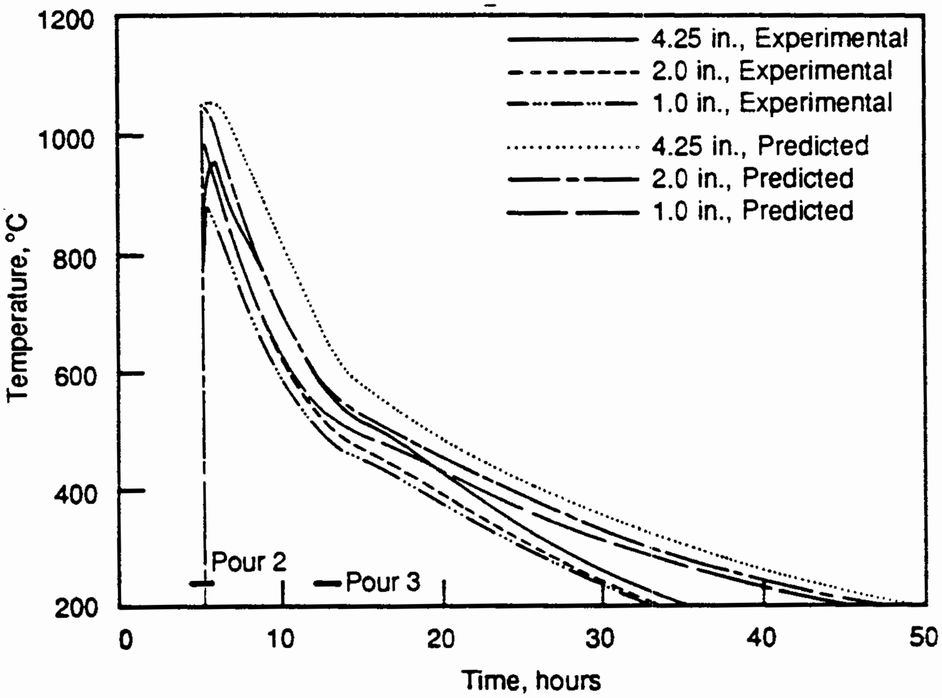


FIGURE 6.19. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 20.0 in. from Canister Bottom

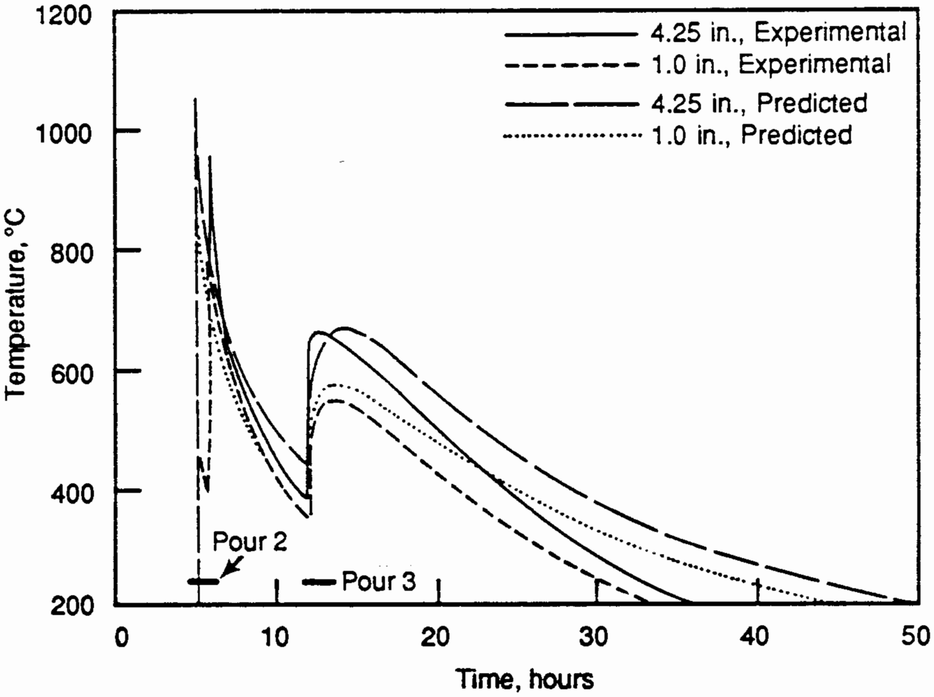


FIGURE 6.20. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 24.875 in. from Canister Bottom

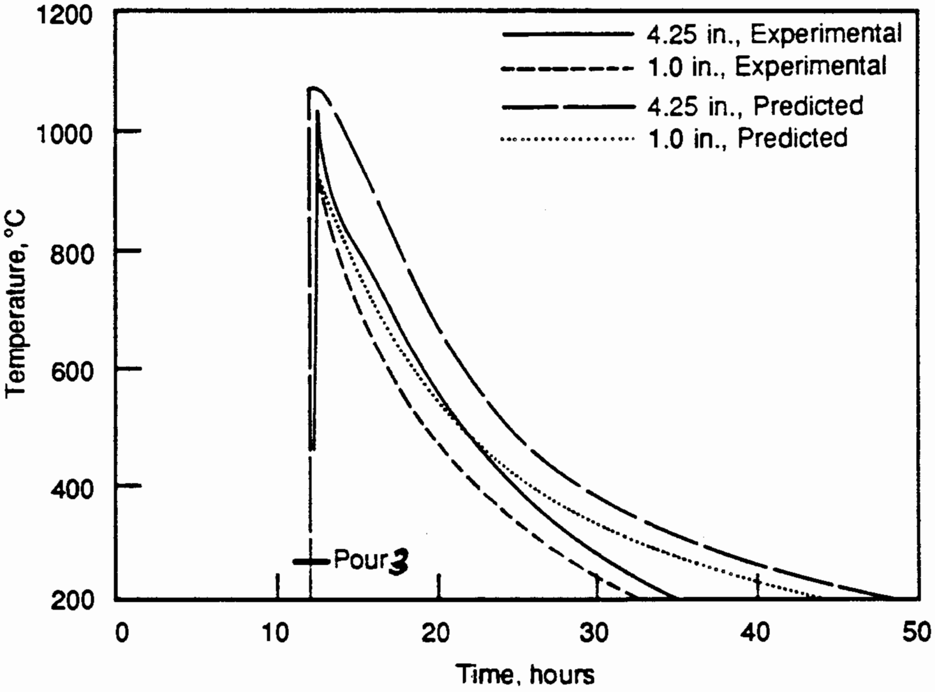


FIGURE 6.21. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 32.75 in. from Canister Bottom

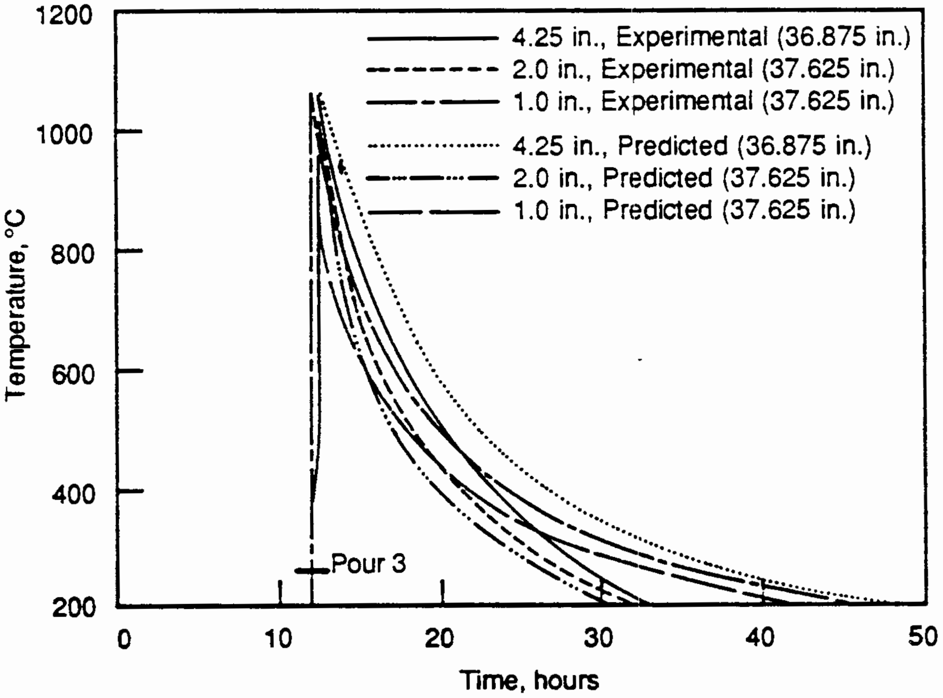


FIGURE 6.22. TEMPEST Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Levels 35.875 and 37.625 in. from Canister Bottom

canisters. However, significant reheating occurred from the second pour at the second and third axial thermocouple levels, as shown in Figures 6.17 and 6.18. As in the case of 12HI, the TEMPEST-predicted cooldown immediately after the first pour was too slow. However, instead of an improvement in agreement as time progressed after the first pour, the difference between the observed and predicted temperatures increased, despite the greater thermal driving force caused by the higher temperatures.

It is also interesting to note that TEMPEST predicted a less steep reheat at the second and third axial thermocouple levels of canister 12LI than that observed (Figures 6.17 and 6.18).

Figures 6.19 and 6.20 show the results for the two axial thermocouple levels covered during the second pour. The high temperatures predicted by TEMPEST for thermocouple level 4 (Figure 6.19) during the period immediately after the pour appear to be caused by TEMPEST using a higher initial glass temperature. However, consistent with the first pour, the predicted cooling rate beyond 12 h was lower than that observed in the experiment. The observed and predicted points of inflection caused by reheating from the third pour appear similar (Figure 6.19).

The fifth axial thermocouple level is 24.9 in. from the bottom of the canister, which is only 0.6 in. from the top of the second pour. Careful analyses of the raw data clearly show that the fifth axial thermocouple level for canister 12LI was covered during the pour. However, glass levels cannot be predicted to better than about an inch. Significant axial temperature gradients exist near the surface because of the large radiant flux there. The agreement between experimental and predicted values (Figure 6.20) is very good, considering the uncertainties. The reheat pulse was reasonably modeled. However, the predicted long-term cooling rate was slower than that observed.

Figures 6.21 and 6.22 present the predicted and observed results for the three axial thermocouple levels covered during the third glass pour in canister 12LI. The TEMPEST predictions immediately after the fill were lower than the observed temperatures instead of being too high. However, as with canister 12HI, the predicted long-term cooling rates were too low. Note that TEMPEST correctly predicts that the temperature of the uppermost axial thermocouple level, which is only 1.9 in. from the surface, is lower than that on the next lowest axial thermocouple level at the same radial position, which is located 3.7 in. from the surface, because of axial cooling affects.

Figure 6.23 presents the predicted and observed axial temperature profiles at the 4.25-in. radial location for canister 12LI. The figure is composed of three sections. The uppermost section gives the TEMPEST predictions (solid line) for the glass in the canister after the first pour. The middle and bottom sections present the TEMPEST predictions for the glass after the second and final (third) pours, respectively. In all cases, the experimental data are plotted as points and the predictions are plotted as curves. The curves are plotted with time as the parameter. The times were selected to give values immediately after a pour was finished, roughly midway between two pours, and immediately before the next pour.

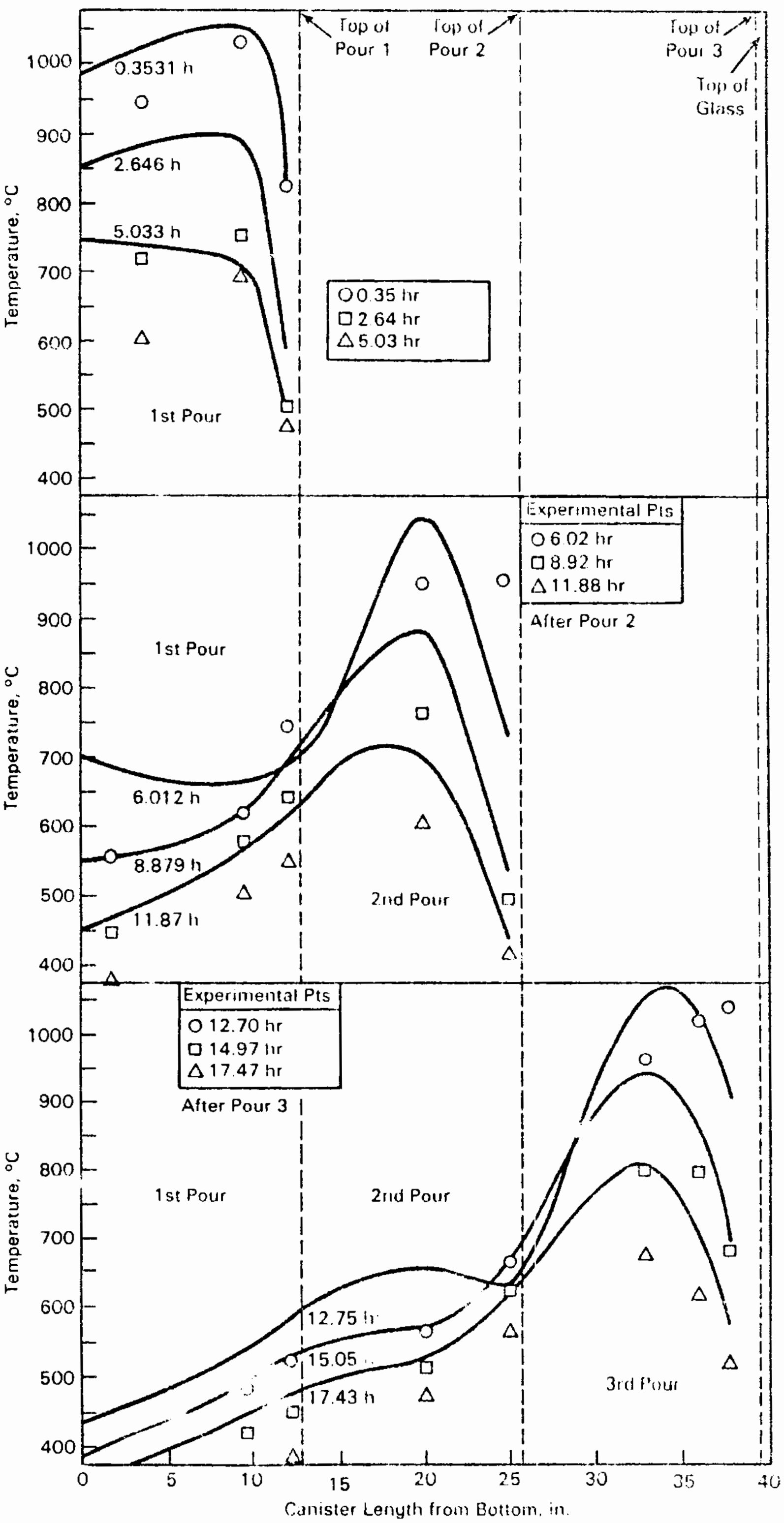


FIGURE 6.23. Axial Profiles for Canister 12LI

In most cases, the TEMPEST predictions were higher than the observed temperatures. A few exceptions occurred at pour interfaces immediately after a pour. For example, note that the data point at 12.125 in. at 6.012 h lies above the TEMPEST prediction line. Note also that the agreement between the experimental points and the TEMPEST predictions was relatively accurate at the glass/air interface for times far removed from the end of the pour.

The predicted axial temperatures near the bottom of the canister, and the predicted shape of the axial profile immediately after the second and third pours (6.012 h and 12.7 h) did not agree closely with the experimental data. The high predicted temperatures at the bottom of the canister are probably due to the assumption made in the TEMPEST batch-fill model, that the glass for each pour is instantaneously placed in the canister. It is also possible that the high predictions for the temperatures at the bottom of the canister may be caused by larger than expected heat transfer out of the bottom of the canister. However, instead of instantaneously filling each canister with glass, the three batch pours required 0.25 h, 0.93 h and 0.82 h, respectively. During these periods, hot, new glass was continuously added to the glass surface, thereby keeping the surface temperature high. The batch model, however, postulates that a hot cylinder of glass, roughly 13 in. high, is instantaneously added to the canister. This entire cylinder begins to cool at the start of the pour from both ends. Thus, the batch-fill model:

- overpredicts the temperature at the bottom of the batch immediately after a pour because it underestimates the heat loss from radiation that occurred at this position during the initial stage of the pour
- underpredicts the temperature at the top of the batch because the model starts cooling the top of the pour at the beginning of the pour, where, in reality, the top of the pour consists of fresh, hot glass placed there at the end of the pour.

The cooling of each batch of glass will result in predicted temperature profiles that peak in the center of each pour; this is seen in all three cases. The predicted temperature profiles for the top of the first batch immediately after pouring (0.35 h) were reasonably accurate, because the pour was relatively short. However, the TEMPEST predictions for the second and third pours were much lower than the actual temperature profiles. This is because of the long periods for pours 2 and 3 (0.93 h and 0.82 h, respectively). The inaccuracies involved in batch-pouring simulations are the primary reason why work was begun on an improved version of TEMPEST, as described in Section 3.3.

Figures 6.24 through 6.26 present comparisons of the observed and predicted temperatures for three axial thermocouple levels at the 4.25-in. radial position. Both the high-temperature and low-temperature radiation models are included in each figure. As mentioned above, use of these two models was expected to bound the radiant heat transfer from the surface if all other aspects of the modeling were accurate. Although the low-temperature model does indeed predict lower glass temperatures, the effect is very small for six of the eight levels. The remaining two levels showed a slightly increased sensitivity to the model used (Figures 6.25 and 6.26). Thus, except for a few relatively short times at two axial levels, the predicted temperatures remained above the observed results in a manner entirely consistent with the high-temperature case.

The reduced temperature sensitivity of the high- and low-radiation heat-transfer models is probably caused by the much higher rate of heat losses from the canister surface. These higher heat losses are caused by the thinner layer of external insulation. As a result, radiation heat transfer from the glass surface is much less important than for the 3-in. insulated canister (12HI). The two exceptions, Figures 6.25 and 6.26, were for thermocouple locations near the top of the pours. Because the effect of radiation cooling is largest at these positions, it is not surprising that larger differences are found.

### 6.1.2 TEMPEST Continuous-Fill Model

As mentioned earlier, the continuous-fill model, TEMPEST Version M, was used to model canisters 13 and 24. Although the latter could have been modeled using the batch-fill version, the large number of batches (33) made the continuous-fill model a better choice because it eliminated the considerable manual labor associated with each restart.

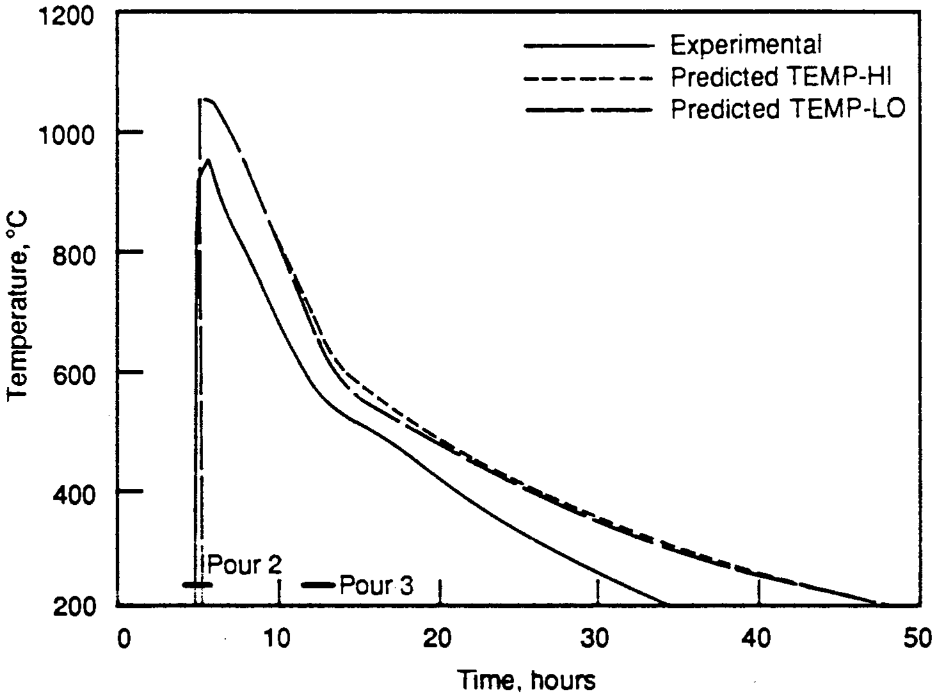


FIGURE 6.24. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 12.175 in. from Canister Bottom

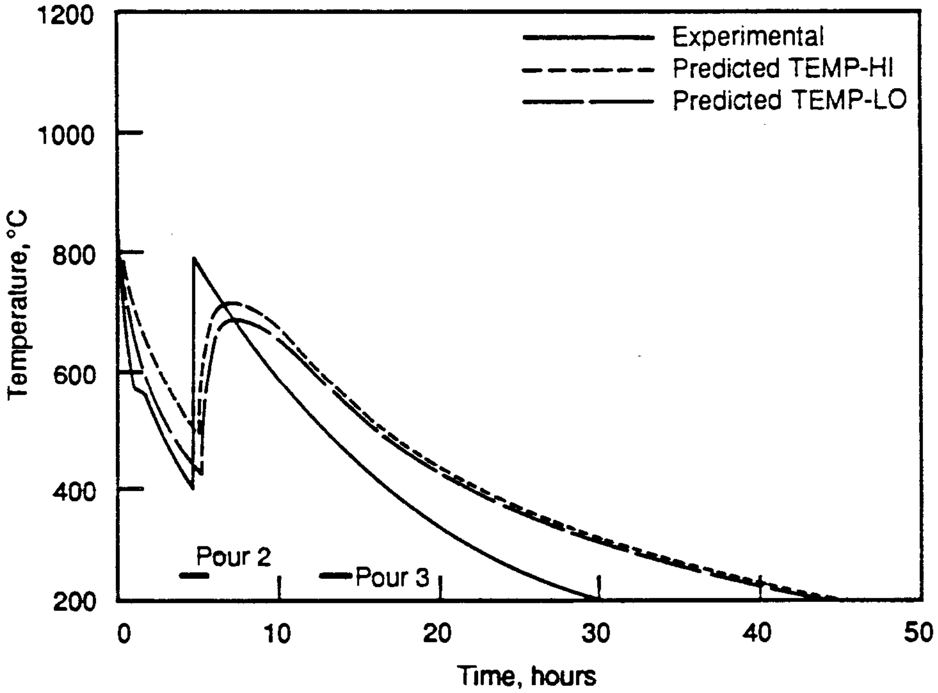


FIGURE 6.25. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 20 in. from Canister Bottom

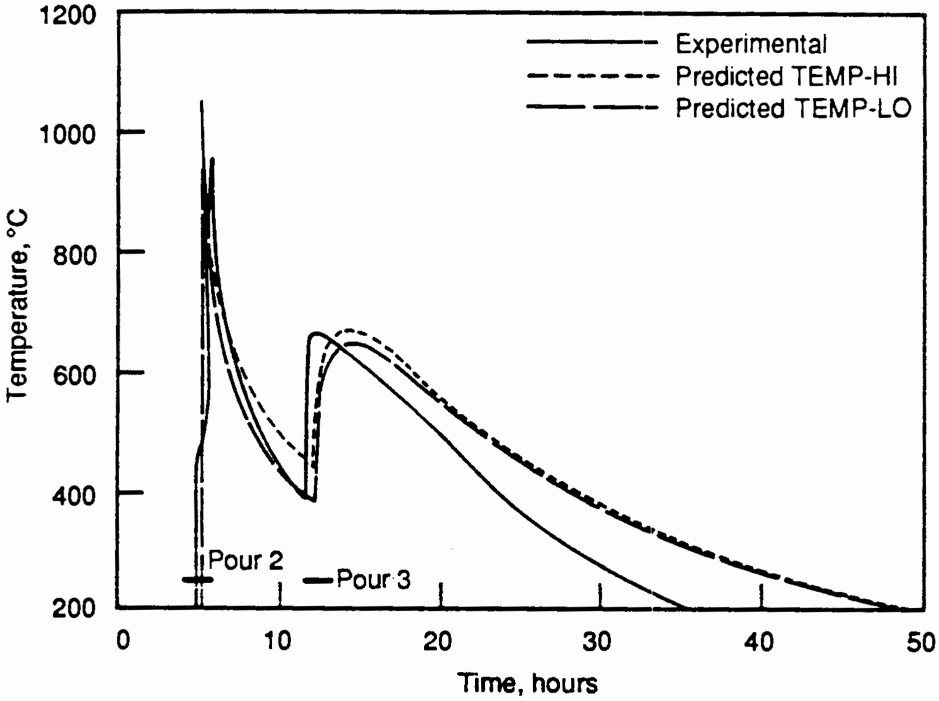


FIGURE 6.26. TEMPEST High- and Low-Temperature Radiation Model Predictions and Experimental Temperatures Measured in Canister 12LI at Axial Level 24.875 in. from Canister Bottom

### 13-in.-Diameter Canister

The TEMPEST model for the 13-in.-diameter canister was insulated with 1/8 in. of Fiberfrax on the inside and 1/4 in. of Durablanket on the outside (see Table 4.1). This is slightly different from the actual canister geometry. The canister was filled using a continuous pouring procedure. The fill rate used was an average value assuming constant fill rates between thermocouple levels during the 263-min period required to fill the canister (see Table 4.10). Figures 6.27 through 6.32 compare the TEMPEST predictions to the observed temperatures for the six thermocouple levels.

The long-term cooling behavior is similar to that found with the batch pours. However, with one exception TEMPEST also predicts a slower cooling rate than that observed after the initial transient (the thermocouple at the glass/Fiberfrax interface; see Figure 6.32). This difference results in the predicted temperatures of each position eventually exceeding the observed temperatures.

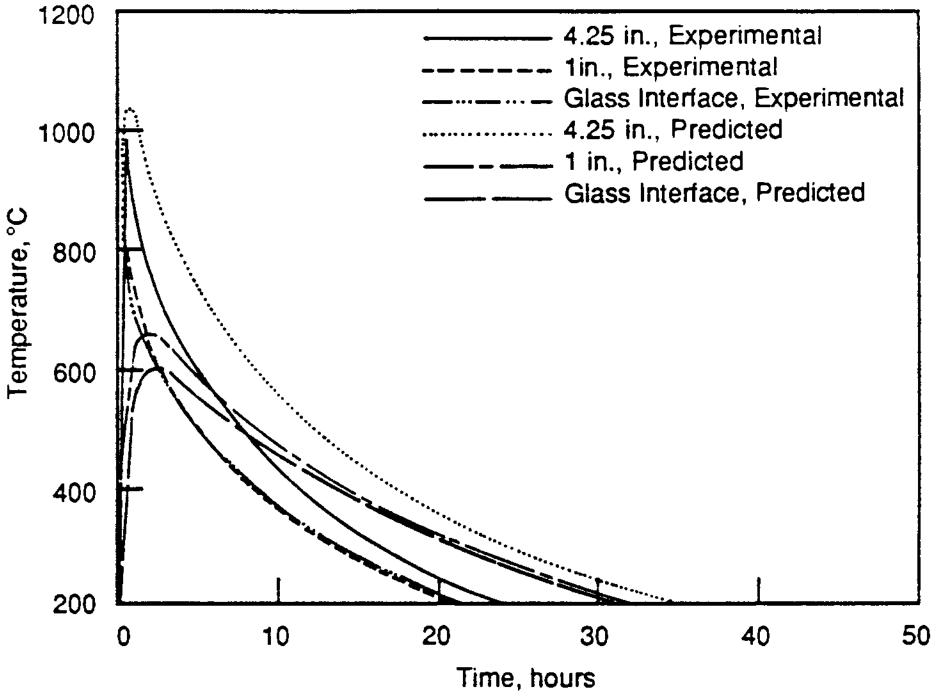


FIGURE 6.27. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 3.5 in. from Canister Bottom

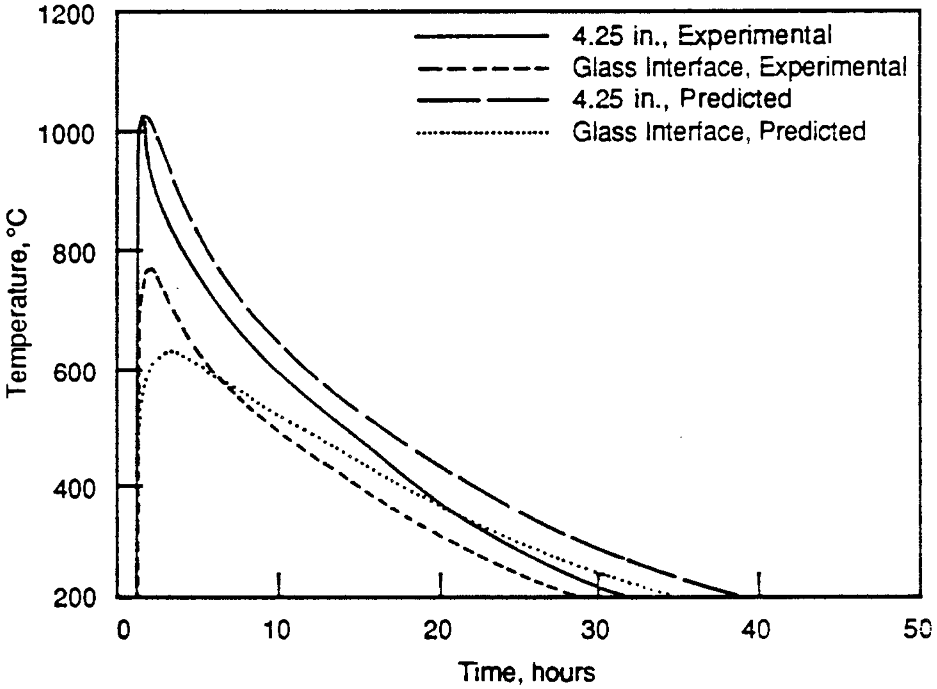


FIGURE 6.28. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 13.3 in. from Canister Bottom

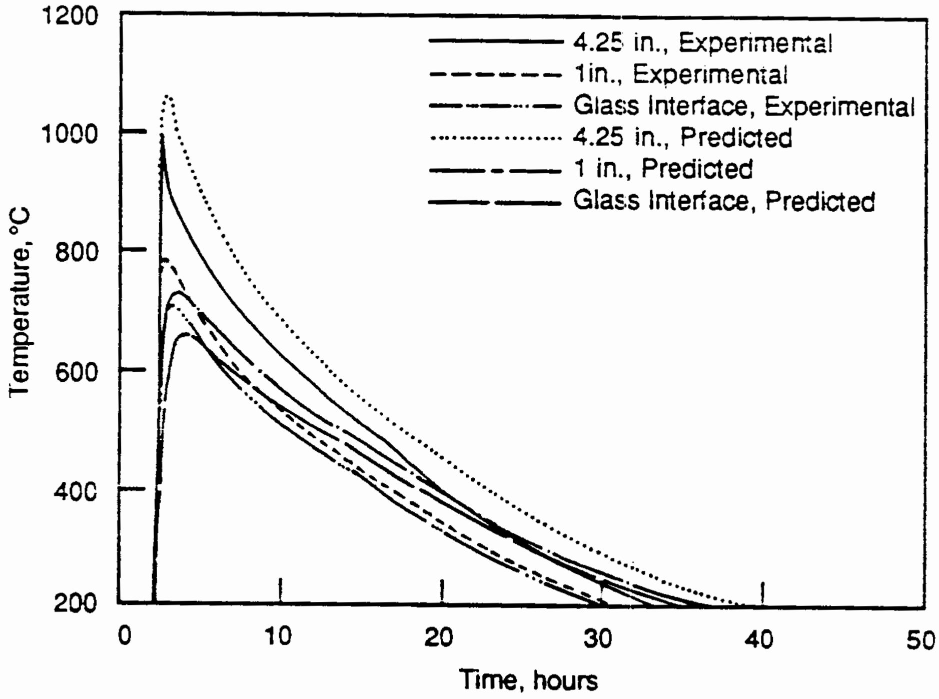


FIGURE 6.29. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 21.1 in. from Canister Bottom

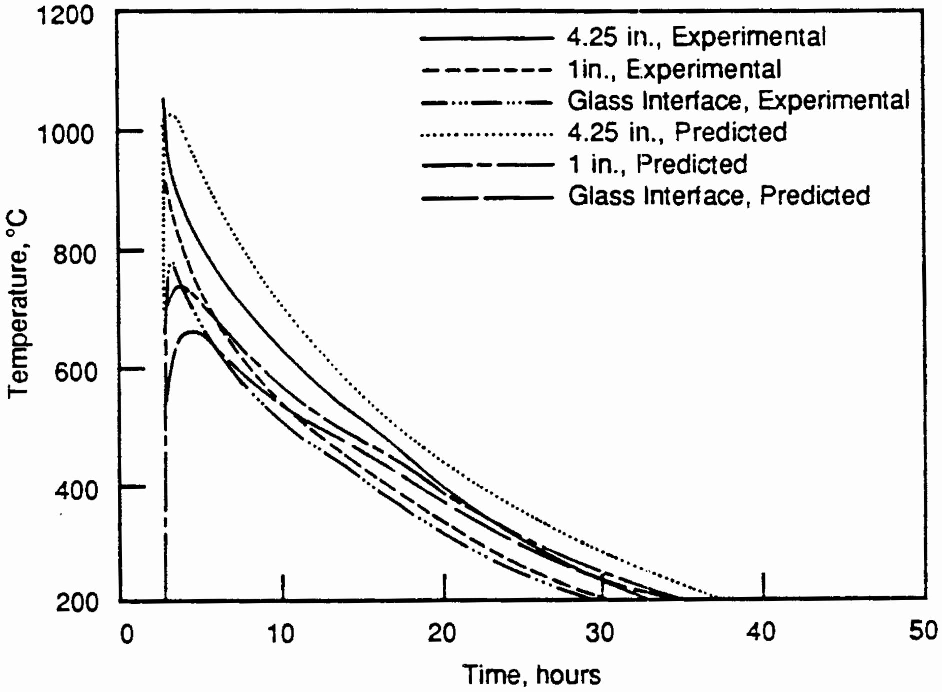


FIGURE 6.30. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 26.1 in. from Canister Bottom

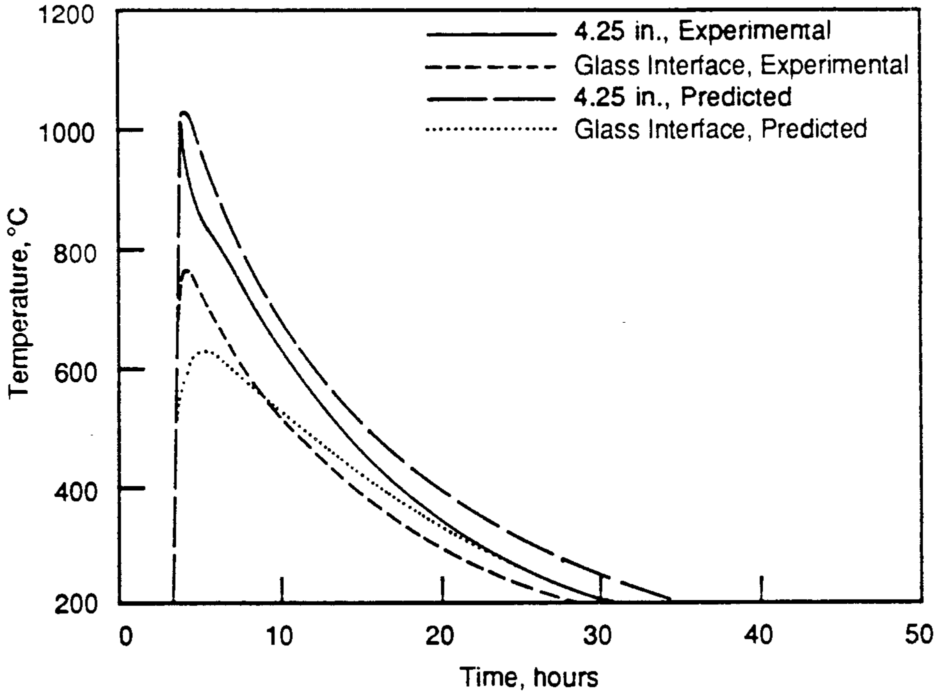


FIGURE 6.31. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 38.4 in. from Canister Bottom

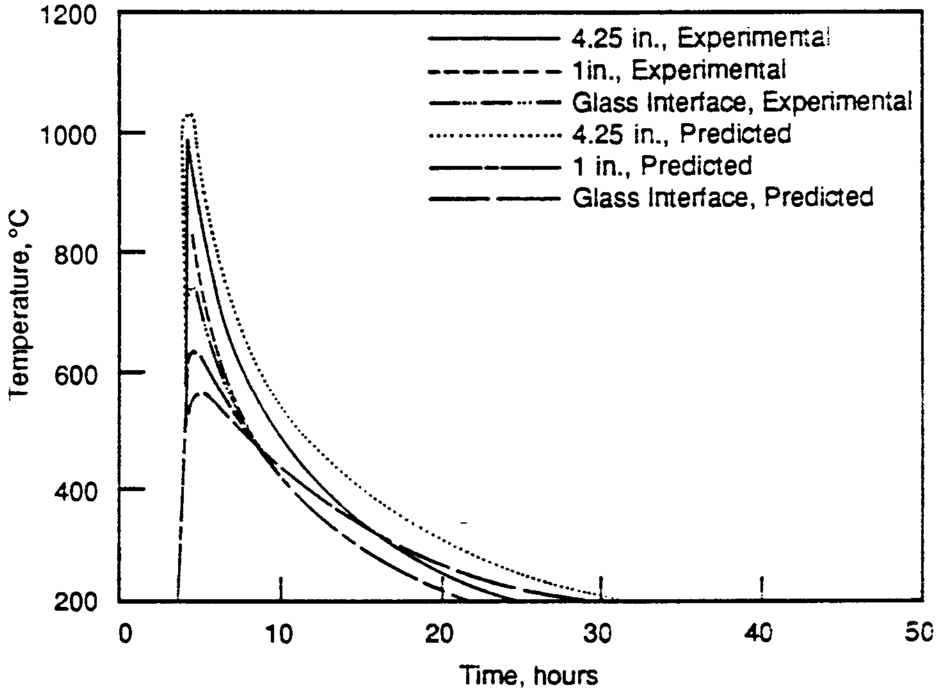


FIGURE 6.32. TEMPEST Predictions and Experimental Temperatures for the 13-in.-Diameter Canister at Axial Level 34.0 in. from Canister Bottom

The cooling behavior that follows the covering of an axial thermocouple level with glass is more complicated. The temperatures of the glass used by TEMPEST were taken from the experimentally observed maximum peak temperature at each level. Thus, the peak temperatures were expected to agree reasonably well at the 4.25-in. radial position. This was generally found to be the case. However, the predicted peaks occurred 15 min later than the observed peaks. Given a mean rate of glass surface rise of 0.15 in./min (40.1 in. of glass poured in 263 min), this 15-min delay is equivalent to 2.25 in. of glass, which is significantly larger than the estimated glass level accuracy of 1 in. The 15 min delay is probably due to the cell dimensioning restrictions, which caused the axial dimension of each cell in the glass filling region to be uniform. The TEMPEST predictions at the 4.25-in. radial position also show a somewhat lower rate of cooling than that observed; however, the agreement is much better than that observed with the batch-fill TEMPEST version used with canisters 12LI and 12HI.

TEMPEST predictions of the maximum temperature for the period immediately following glass coverage of the 1-in. radial thermocouples occur 30 to 90 min later than observed, with the exception of the uppermost level, which was only 15 min later (see Figure 6.32). In addition, the predicted peak temperatures range from 50°C to 400°C lower than those observed. This is probably caused by the inviscid convection assumption of the continuous-fill model, which neglects increases in viscosity of the glass as it cools. This results in cooled glass beneath the TEMPEST glass surface cell being forced out to the wall.

TEMPEST predictions for the transient following coverage of the thermocouples at the glass/Fiberfrax interface were usually 30 min to 3 h late. The predicted peak temperatures were also 50°C to 400°C lower than those observed.

### 24-in.-Diameter Canister

The 33 batch pours for the 24-in.-diameter canister proved to be too difficult to model with the continuous-fill model of TEMPEST. This was caused by the code's inability to handle continuous flow over the wide range of batch pour rates encountered. As a result, the temperatures from the canister modeling could not be compared to the actual glass temperatures in the canister.

## 6.2 TURNTABLE SIMULANT EXPERIMENT

The experiment in which canister 12TTS was filled in the turntable simulant was conducted for three reasons:

- to determine the effects of the asymmetrical thimble structure on circumferential temperature gradients at the canister surface
- to determine the correctness of the assumed emissivities and radiant heat transfer modeling simplifications used in the TEMPEST simulation
- to determine the correctness of assumed contact coefficients simulating convective heat transfer in the TEMPEST simulation.

Results addressing each of these points are summarized below.

The need to examine the effects of asymmetrical emplacement of the thimble in a turntable is due to the requirement by the TEMPEST code for axial symmetry with cylindrical geometries. Symmetrical modeling of the unsymmetrical turntable geometry will give accurate results only if the circumferential temperature gradients at all axial levels are negligible.

To better examine these potential gradients, an extensive three-dimensional circumferential array of thermocouples was placed on the surfaces of each canister or turntable shell. The circumferential temperature gradients at various axial levels of the canister were analyzed to determine the existence of these gradients over time. In general, 40°C circumferential temperature gradients occurred on the canister shell immediately after each batch fill. However, this gradient decreased to less than 10°C after the next glass batch was added to the canister. Furthermore, the significant portion of the circumferential gradient occurred over less than 25% of the turntable shell. At a canister temperature around 400°C, the effect of the circumferential gradients on the average temperature at that level is a reduction of approximately 1 to 5°C. On the basis of these observations, it can be assumed that cylindrical modeling of the RLFCM turntable via the TEMPEST code, ignoring circumferential temperature gradients, is an acceptable approximation from a heat transfer perspective.

The other reason for the turntable simulant evaluation stems from the inability of the TEMPEST code to exactly model transient convective and radiative heat transfer concurrently. To estimate the actual convective and radiative heat transfer with TEMPEST, a solid, nonconvecting form of an "infinitely conductive" material, with density and specific heat properties equivalent to air, was used. Radiant heat transfer, along with contact coefficients determined to simulate gap convection, were then placed between each metal shell and the "infinitely conductive" material to simulate the heat transfer between shells. The effectiveness of this simplification was evaluated.

A steady-state computer simulation of the turntable simulant was made by TEMPEST. The experimentally obtained canister shell temperature profile was taken from a point in time where the cooling rate of the canister had slowed and the glass at the hottest point on the glass/Fiberflax interface was just beginning to harden. This condition was chosen because it was at a temperature in the primary range of interest at a time when the rate of change in temperature was low enough to make a reasonable approximation possible. The axial canister temperature profiles that were input into TEMPEST varied from 234°C at the top of the canister wall, to 411°C at canister mid-level, to 272°C at the canister bottom. Emissivities of 0.7 for the canister surface and 0.5 for all other shells were used. The emissivity values were based on values given for appropriate stainless-steel oxidation levels and were estimated to be the highest possible emissivities, based on the maximum experimental temperatures of each of these shells. The emissivities used should result in faster cooling rates and larger radial temperature gradients than would actually exist. The contact coefficients used to simulate convective heat transfer were determined using steady-state convective heat transfer analyses.

One assumption required for comparing steady-state analysis with transient cooling profiles is the presence of insignificant lag time in temperature response between shells. A comparison of the metal shell temperatures over time shows a "lag time" of approximately 3 h between a batch fill in the canister and the complete temperature response of the turntable shell. However, much of this lag time occurs primarily in the outer shells where the temperature variation over time is slow. Observation of the transient cooling and reheating profiles of the metal shells shows a maximum expected error of 25°C at each metal shell, if a steady-state analysis at this point in time is used.

Due to the modeling simplification required by TEMPEST, the computed axial temperature gradients on metal shells in the turntable simulant were 50 to 200% greater than the actual profiles observed in the run. To better examine our assumptions, the average temperatures of the experimental and TEMPEST-simulated cooldowns were computed for each metal shell. The difference in average temperatures between adjacent shells were then determined, and comparison was made between the experimental and computed temperatures. The results of this comparison, shown in Table 6.1, indicate reasonably good agreement between the experimentally-derived and the TEMPEST-predicted temperatures. The only discrepancies occur in the region between the canister shell and overpack and the region between the turntable wall and ambient. Attempts to improve these discrepancies are discussed below.

Improved convective heat transfer calculations were performed based on experimental shell temperatures; these calculations resulted in variations of less than 8% in the total amount of heat transfer. To examine the effect of

TABLE 6.1. Comparison of Computer-Predicted Temperature Drops with Experimental Results Between Shells of the PSCM Turntable Simulant

| | Average Temperature Drops | | |
|---|---|---|---|
| Gap Location | Experimental | Computer Pred.[a] | Lowered Emissivities[b] |
| Canister wall to overpack | 70°C | 48°C | 52°C |
| Overpack to seal liner | 80°C | 75°C | 79°C |
| Seal liner to thimble | 64°C | 72°C | 76°C |
| Thimble to turntable wall | 64°C | 67°C | 61°C |
| Turntable wall to ambient[c] | 19°C | 35°C | 28°C |

(a) Canister emissivity 0.7; other shell emissivities 0.5.
(b) Canister emissivity 0.5; other shell emissivities 0.4.
(c) Ambient air temperature = 22°C.

improved convective heat transfer, along with variations in emissivity, another steady-state calculation was made with improved convective heat transfer coefficients, a canister emissivity of 0.5, and all other shell emissivities of 0.4 (see Table 6.1, Column 3). The results of this simulation are similar to the higher emissivity simulation, with noticeable improvements. This suggests a flatter emissivity profile than expected. However, discrepancies still occur in the temperature shell'a axial distribution. These findings indicate that emissivities are not the only reason for these discrepancies. Other modeling concerns that may affect the comparison of the computer simulations with experimental results are:

- thermocouple error in the experimental results
- limitations and inaccuracies in the radiant/convective heat transfer modeling required by the code
- limitations in the use of steady-state analysis to predict transient cooling temperatures between shells.

In summary, the use of a turntable simulant demonstrated that the TEMPEST code can effectively simulate the heat transfer resistance caused by emplacement of a canister in the RLFCM turntable. Although some discrepancies exist, the agreement is close enough to give a reasonably accurate prediction of the radioactive canister's cooling rates and temperature profiles.

# 7.0 DISCUSSION

Four main topics are covered in this chapter. The results obtained with the TEMPEST batch-fill model are discussed in Section 7.1. In Section 7.2, the TEMPEST continuous-fill model results are explained. Next, the implications from the turntable simulant results are presented in Section 7.3. The chapter concludes with Section 7.4, which provides a general discussion of problems encountered in using the code, opportunities for using the code, suggested code development activities, and potential input model and associated model activities.

## 7.1 TEMPEST BATCH-FILL MODEL RESULTS

When used to model the batch-fill experiments, the TEMPEST code did a reasonable job of predicting temperatures. The maximum deviations from experimental results (100 to 250°C) occurred during the initial cooldown transient. Thereafter, the TEMPEST predictions were generally within 125°C of the observed temperatures. These latter deviations would be smaller if it were not for the code's failure to model the initial cooldown effect (see Figure 6.4, for example). Two factors account for most of the deviations:

- a large underprediction of the cooling rates for the 8 to 10 h immediately after pouring
- a less pronounced underprediction of the cooling rates for times longer than 10 h after pouring.

Possible reasons for the existence of these factors include:

- changes in conductive heat transfer that are caused by glass fracturing
- inaccurate modeling of radiant heat transfer from the glass surface
- errors implicit in the assumption that the entire batch of glass added during a pour is instantaneously dumped into the canister
- the assumption that only conductive heat transfer occurs in the glass

- errors in the code-required input for physical properties of glass at high temperature
- an overestimation of the insulation properties of Fiberfrax and/or Durablanket.

Each of these reasons is discussed below.

Significant glass cracking with concurrent formation of gaps could conceivably decrease the effective thermal conductivity of glass. However, this explanation does not seem plausible for canisters 12HI or 12LI, for three reasons. First, for a given temperature at the 4.25-in. thermocouple (chosen because it is closest to the center), the radial temperature differences between the 4.25-in. and 1-in. thermocouple are very close to the observed temperature differences. Second, the temperature differences within the glass are far smaller than they are between the glass surface (whether on the bottom, sides, or top) and the ambient temperature. Even if cracking had increased these gradients to twice that of crack-free glass, the effect on temperature drop would have been far smaller than the deviations between TEMPEST and the observed results. Third, a decrease in actual thermal conductivity of glass would serve only to decrease experimental cooling rates, not increase them as observed. Therefore, we conclude that glass cracking is not a significant cause for the observed deviations.

As discussed in Chapters 3 and 5, TEMPEST is incapable of modeling radiation and convection heat transfer concurrently in an exact manner. This undoubtedly leads to errors in predicted temperatures. However, for the canister filling conditions encountered in this study, the radiation model used appears to cause relatively minor errors. This is evidenced by the small differences in the predicted temperatures when using the high-temperature and the low-temperature radiation models (see Figures 6.8 through 6.15 and 6.24 through 6.32). Note that the differences in the predictions made by these two models are far smaller than the differences between the predictions of the models and the observed results. An improved radiation model should be developed to ensure that this potential source of error does not become significant for other TEMPEST applications to LFCM heat transfer calculations.

The errors associated with assuming that the entire amount of glass poured during a batch is instantly placed in the canister is the probable cause of much of the slow cooldown prediction by TEMPEST immediately following a pour. If this source of error were to be eliminated, the agreement between the TEMPEST predictions and the experimental results would be dramatically improved. Evidence supporting this hypothesis is found in the consistently superior predictions obtained using TEMPEST Version M (the continuous pour model) in the 13-in.-diameter canister. This is discussed in more detail in Section 7.2.

To understand the cause for the errors implicit in the batch version, it helps to consider the fill process. Batch filling gets its name because it is discontinuous. The LFCM is allowed to fill; the canister is then filled at a substantially faster rate than would be possible under continuous filling because the latter is limited by the glass production rate. However, the duration of the fill, which ranged from 23 to 54 min for canisters 12HI and 12LI, is still long enough to permit significant heat transfer. The batch fill version partially accounts for this by adding the filling time to the cooling time between pours so that total cooling time remains the same. However, the batch-fill version does not accurately model the temperature of the glass surface during the filling process. In reality, a stream of glass approximately 1/6 in. in diameter strikes the glass surface; the point of impact meanders about the center of the canister as a result of wetting and Coanda effects at the pour spout of the LFCM. This glass stream partially submerges itself in the glass. Pressure gradients set up from the momentum of this stream, buoyancy effects due to convection, and gravity effects due to the glass impacting near the center cause the glass surface to move upward, maintaining itself level perhaps to within an inch or so. This source of new, hot glass keeps the surface at an elevated temperature that is determined by the rate at which heat is added from the new glass and lost by radiation and convection from the glass surface.

In the TEMPEST batch pour model, the entire pour (about 13 in. for canisters 12HI and 12LI) enters the canister instantly. This glass is cooled at the top, bottom, and sides. At the top, the glass surface is cooled by radiation and convection as described above. However, the modeled surface temperature at the top of the canister will fall much more rapidly than the actual temperature because the surface is not being renewed with new, hot glass. The lower surface temperatures will cause a much lower heat transfer rate from the top of the glass (because radiation, the dominant heat transfer mechanism soon after a pour, is proportional to temperature to the fourth power). This lower heat transfer rate probably accounts for much of the early post-cooldown deviation.

The assumption of solely conductive heat transfer in the glass may account for some of the remaining deviation in early post-cooldown temperatures. The batch-fill model assumes that the glass is a solid, that is, no convective heat transfer within the glass is assumed, so only conduction is computed. Examination of the experimental data shows isothermal regions near the glass surface. These regions are at least 2 in. deep by 4 in. across, which indicates extremely high rates of heat transfer. The TEMPEST simulations show significant temperature gradients in the same region, hence lower heat transfer. Clearly, a more accurate modeling of the region immediately below the surface would improve the accuracy of the TEMPEST predictions.

One possible explanation for this apparently extremely high rate of heat transfer is that the glass in these regions (~1000°C) may be sufficiently fluid that a combination of convective cells, combined with stirring caused by the impacting glass stream, increases heat transfer far above that predicted by conduction. Another possibility is that the values of high temperature thermal diffusivity used by canister 12HI and 12LI are in error. The technique used to measure glass thermal diffusivity is well accepted for solids; its applicability to molten glass needs to be verified, however.

Finally, there is a significant underprediction of cooling rate for long periods after each batch pour. This underprediction is manifested by the lower rates of cooling exhibited by TEMPEST later in the cooling process see Figures 6.4, 6.6, 6.16, 6.17, and 6.20). The underprediction occurs even though the predicted glass temperatures are higher than measured, which should cause a higher cooling rate. The most likely explanation for the difference is that the actual thermal conductivities for Fiberfrax and/or Durablanket insulation are higher than that used for the TEMPEST simulations. Several reasons support this explanation. First, the TEMPEST predictions all err in the same direction for the later portions of the cooldown process. Second, and perhaps more convincing, TEMPEST exhibits this deviation for the case of canister 12HI. Canister 12HI was insulated with 1/8 in. of Fiberfrax internally and 3 in. of Durablanket externally. Therefore, the limiting thermal resistance should be the Durablanket. Examination of Figures 6.1 through 6.7 shows that the 4.25-in. radial temperatures at 50 h range from 420°C at the bottom to 630°C at level 5 and back down to 525°C at level 8. The temperature differences between the 4.25-in. and 1-in. thermocouples in the glass are quite small at 50 h, however, ranging from 0 to 50°C. Temperatures at the 1-in. thermocouple locations at 50 h range from 400 to 600°C. Unless there is an extremely large temperature drop in the center 1 in. of glass, well over 80% of the temperature drop, hence thermal resistance, is in the Fiberfrax and the Durablanket, of which the latter provides by far the greater resistance.

Observations show that the predicted time/temperature slope at 50 h is either equal to or slightly greater than that of the observed results, even though the predicted temperatures are about 100°C higher. This 100°C higher temperature translates into approximately 25% greater temperature driving force to the ambient. Thus, it is tentatively concluded that the effective thermal conductivity of the Durablanket/Fiberfrax insulation was about 25% higher than that used in the TEMPEST input.

Further analyses, which would include using the thermocouple data collected at the surface of the canister, could determine if some unforseen heat transfer resistances exist inside the Durablanket. If no such resistances are found, the problem is probably due to the thermal conductivity values given for Durablanket; the heat transfer calculation for canister 12HI after 50 h is a simple conduction problem well within the verified capabilities of TEMPEST.

## 7.2 TEMPEST CONTINUOUS-FILL MODEL RESULTS

The TEMPEST continuous-fill version was used to model the 13-in.-diameter canister. The long-range cooling behavior predicted for the canister is similar to that found with the batch-fill model results described above. The most likely explanation is the same: a 25% underprediction of the insulating values of Durablanket (and/or possibly Fiberfrax).

Canister 12LI and the 13-in.-diameter canister are similar in the amount of insulation and in size. In spite of these similarities, it is difficult to make direct comparisons between their cooldown rates because the 13-in.-diameter canister was filled in only 4 h, whereas canister 12LI was filled in three pours over a 12.7-h period. However, spot checks made at similar thermocouple levels consistently showed much better agreement between the predicted temperatures and the observed temperatures for the 13-in.-diameter canister. Except for the lowest thermocouple level, where the deviation was similar (125°C for canister 12LI compared with 110°C for the 13-in.-diameter canister), deviations for the 13-in.-diameter canister were about half those of canister 12LI. This is probably the result of better predictions of the transient by the continuous-fill model, which accounts for the replenishment of the surface layer with hot glass (Section 7.1). In short, over most of the cooldown, the TEMPEST predicted temperatures that were within 50°C of those observed for most of the thermocouple locations in the 13-in.-diameter canister.

However, significant differences in short-term predictions of the temperature gradients occurred as a function of radial position. TEMPEST was late in predicting the peak temperatures. This error increased with increasing thermocouple distance from the canister center, reaching 3 h at the canister wall. TEMPEST also predicted peak temperatures that were too low. Again, this error increased with distance from the center, with errors reaching 400°C at the glass/wall interface. This deviation is consistent with the position taken in Section 7.1: TEMPEST appears to grossly underpredict the rate of heat transfer near the glass surface, either because of the use of inaccurate thermal properties, or because mixing, convection, or some other phenomenon is greatly increasing heat transfer above that predicted by conduction alone. Another potential reason is the large axial cell dimensions in the glass filling region.

It should be noted that the advanced N29 version of TEMPEST is designed to rectify these deficiencies. In this version, the glass/air interface will be tracked more exactly and the mesh will be rezoned at each time step. Furthermore, the user will be allowed to specify how many and what size cells he wishes to use to resolve the front. This option will allow automatic and dynamic mesh refinement at the interface, precisely the location of maximum temperature gradients and highest temperatures. Thus, the radiation model will automatically be improved by resolving the high temperature gradients at the glass surface rather than smearing them over too large a cell.

## 7.3 IMPLICATIONS OF THE TURNTABLE SIMULANT RESULTS

Generalized results of the turntable simulant analysis show that the primary modeling problems stem from insufficient knowledge of how stainless-steel emissivity changes over time with temperature and from the inability of the TEMPEST code to simultaneously model radiation and convection between the steel shells. Again, completion of the N29 version of TEMPEST should partially rectify this problem. Although the results show a circumferential temperature gradient on the canister shell caused by asymmetrical emplacement in the turntable simulant, the gradient is only 40°C or less and is, therefore, of secondary concern. A discussion of the two main areas follows.

The way in which the canister and turntable shells change their surface emissivity as a function of temperature and time becomes increasingly important at higher canister surface temperatures. Currently, there is insufficient information on how fast canister emissivity changes, and what the maximum emissivity is for various shell temperatures. Arrowsmith et al. (1977) have studied spectral emissivity in the infrared region (16 to 27 μm) at temperatures up to 600°C. Their studies demonstrate no appreciable oxide formation and thus no significant change in emissivity for Hastelloy C-276 tubing exposed to air for 16 h at 500°C, or 1 h at 600°C. The emissivity for unoxidized Hastelloy tubing is shown to be around 0.25. Other information, however, shows the actual canister emissivity rising above 0.5 at 500 and 600°C, causing over a 100% increase in radiant heat transfer. More detailed descriptions on 304L stainless steel emissivity and its variation over temperature would be desirable.

As described in Section 3.0, radiation between canister shells is modeled by placing in the gap a material that is highly conductive radially and adiabatic axially. This method allows the surface temperature of the canister shells to radiate to each other. A combined radiant and convective heat transfer coefficient is then placed between the shell and material to attempt to simulate heat transfer across the gap. However, this simplification does not handle axial heat transfer, particularly convection and all but direct radiation, across the gaps. This inadequacy is evidenced by the modeled axial temperature profiles along each shell. The profiles show high temperature at the bottom, top, and middle of each canister shell, and low temperatures in between. The high temperature in the middle of the shell occurs near the same level as the hottest glass temperatures, and the high temperatures at the top and bottom of the canister are caused by conduction from the inner and outer shells. This pattern is entirely different from that of the profiles generated in the actual experiments. Temperatures in the actual experiments tend to be hotter in the upper half of the shell and cool off without further temperature increase on the ends. Because of the significant difference between experimental and modeled temperature profiles, two-dimensional heat driving force on the shells of the turntable simulant is being generated to incorrectly model heat transfer off the canister. Code development of the N29 version to more correctly model radiant and convective heat transfer across shell gaps would greatly improve future turntable simulant analyses. A heat transfer measurement during the turntable simulant analysis would also help.

## 7.4 TEMPEST CODE DEVELOPMENT CONSIDERATIONS

The amount of time spent learning and understanding the TEMPEST code, revising it to fit the canister input models, and running the computer simulation was quite extensive. This is because the code is currently not written to handle canister filling and heat transfer modeling. Development of a user-friendly code and manual for heat transfer modeling of canister filling and cooldown would be highly desirable, particularly for the continuous-fill version of the TEMPEST code, which currently has no documentation describing the way it handles a canister fill. Without this development, it is expected that modeling of heat transfer in glass-filled canisters will continue to be tedious and lengthy, minimizing the value of the modeling effort.

In general, the code seems to perform reasonably well when modeling cooling of a canister outside of any turntable environment. It is suspected that experimental uncertainties contributed to some of the difference between the observed and predicted results. Improvements in experimental data recording relating to glass pour temperatures, pour level measurements, and canister surface emissivities (with respect to time and temperature) would be highly desirable. More accurate knowledge of temperature-dependent heat transfer properties for the various materials used in and around the glass-filled canister would also be desirable. Some ideas for improving the experimental data base are listed below:

- During canister filling and cooldown, surface emissivities at various positions on the canister should be recorded over time. An attempt to find a way to determine the temperature of the glass pour stream along with the pour rate during each fill should also be made. This could be accomplished by more frequent recordings of the poured glass temperatures during filling and by using improved methods of determining the glass level after each pour.

- An experiment should be performed to determine the change in emissivity for 304 L stainless steel as a function of temperature from 100 to 1100°C. This table could then be input to TEMPEST to model a continually changing surface emissivity on the canister wall.

- The convective heat transfer of viscous glasses at temperatures above 900°C should be determined. A table of temperature and viscosity versus convective heat transfer could then be developed and input into TEMPEST for more accurate modeling of heat transfer above 900°C.

As stated in Section 3.0, the TEMPEST code was not written for canister heat transfer analysis during glass filling and cooldown. To attempt to model these canisters, the L4 version of the code was revised to handle radiation and heat transfer during filling while trying to decrease the computer time required for each simulation. The two revised versions (L4X and M) performed well considering possible inaccuracies in the input data base. However, development of these code versions represents only a small portion of the code development effort needed to accurately model heat transfer.

Continuous-Fill Modeling - The continuous-fill version of TEMPEST predicts glass temperatures better than the batch-fill version. This conclusion implies that future development efforts should use the continuous-fill model, even in batch-filled canisters. However, the continuous-fill (M) version of the code is currently hardwired to handle only certain sizes of canisters, and no documentation exists for changing the input. In addition, the method for inputting time-dependent glass pours and initial glass temperatures requires much preliminary determination. Development money should instead be spent to finish the N29 version of the TEMPEST code. In the N29 version, fill rates and glass temperatures are input instead of the percentage fill rate in the radial center node and the volumetric amount of sensible heat in the glass (automatically calculated in the N29 version). In addition, the N29 version allows arbitrary, user-selected radial and axial cell sizes throughout the entire mesh, including the glass-filling area. This flexibility allows more exact monitoring of the thermocouple locations. Therefore, completion of the N29 version of TEMPEST would provide the continuous-fill modeling capability desired.

Radiation - Analysis of the 12- and 13-in.-diameter canisters shows that the current models in TEMPEST for computing radiation from the glass surface are a reasonable first try at modeling radiation. However, improved radiation heat transfer models, coupled with improvements in the computation of convective heat transfer across shells, should be developed. These improvements are particularly crucial when attempting to model large-gap radiation between steel shells in a turntable environment. The improvements would also improve the accuracy of the heat transfer calculation for the rising glass surface during a pour. Improved radiation modeling would allow for use of the improved convection capabilities designed for the N29 version of TEMPEST.

Slow Transient Convection Analysis - In both the batch-fill (L4X) and continuous-fill (M) versions of the code, the momentum equation was entirely explicit, mandating very small time steps for fluids like glass with large viscosities. Thus, the L4X and M versions were not able to adequately model the glass convection in the canister. For the models run with the L4X version (canisters 12LI and 12HI), the glass was assumed to be a conducting-only medium. For the models run with the M version of the code, the glass was assumed to be inviscid. The N29 version of the code has been entirely rewritten, and the viscous part of the momentum equation is implicit. Thus, large time steps can be taken with the advanced N29 version of the code so that convection can be modeled economically, even for the type of slow transients involved in canister cooldown calculations.

Asymmetrical Cylindrical Analysis - As stated in Chapter 3, the TEMPEST code cannot currently model asymmetrical emplacement of the turntable shells in an LFCM turntable. Model developers for TEMPEST are currently improving the code's ability to handle noncylindrical coordinate systems, which may help the modeling effort for the asymmetrical turntable geometry. However, this development effort may be as long as 2 years away from completion. Because the turntable simulant analysis shows only a small circumferential gradient on the canister due to asymmetrical emplacement, any development effort in this area can be postponed.

Other developmental efforts to improve TEMPEST modeling of canister temperatures include:

- modeling of the heat transfer of the pour stream as it falls into the canister
- modeling the time- or temperature-dependent emissivities from canister shells.

# 8.0 QUALITY ASSURANCE AND TRACEABILITY

The computer code development work was done according to PNL-MA-70, Impact Level II standards. The base version of TEMPEST is designated to the L4 version and has been sent to the National Energy Software Center in Argonne, Illinois. The installation of the implicit viscous diffusion capability resulted in a complete rewrite of the code. Thus, the N29 version was created to serve as a new base version (designated N29A0) of TEMPEST. It was successfully tested on the standard TEMPEST test problems (Eyler, Trent, and Budden 1983[a]). An update module (N29B0) was then created to allow implicit viscous diffusion (see Appendix A). These update additions plus the N29A0 version were combined to create the N29B1 version. The N29B1 version served as the starting point for the FY 1987 work. When the front tracker module is finished and added to the existing version, it will be designated as N29B2 (see Appendix B). Traceability and configuration management are accomplished by using the software utility HISTORIAN (see Appendix C).

All versions of TEMPEST described in this report were run on the Los Alamos National Laboratory (LANL) Integrated Computer Network (ICN), and stored on their Common File System (CFS). The ICN is composed of various computers; this work was performed in the open partition, which currently comprises one CRAY X-MP and three CRAY-1S machines. All machines run the Cray Time Sharing System (CTSS) operating system. Although the L4X and M versions of TEMPEST were run on CDC-7600 computers running the Livermore Time Sharing System (LTSS) operating system, TEMPEST is written in FORTRAN and is very portable to any machine with a FORTRAN compiler. The TEMPEST N29 version is running on the LANL CRAY-1S is and on the PNL DEC VAX 11-780 computer running VMS 4.3.

[a] Eyler, L. L., D. S. Trent, and M. J. Budden. 1983. TEMPEST - Three-Dimensional Time-Dependent Computer Program for Hydrothermal Analysis. Volume II: Assessment and Verification Results. Pacific Northwest Laboratory, Richland, Washington.

## 9.0 REFERENCES

Arrowsmith, H. W., and R. P. Allen. 1977. Electrical Decontamination of Strontium Fluoride Storage Capsules. BNWL-2125, Pacific Northwest Laboratory, Richland, Washington.

Chikalla, T. D., and J. A. Powell. 1982. Nuclear Waste Management Semiannual Progress Report. PNL-4250, Pacific Northwest Laboratory, Richland, Washington.

Eyler, L. L., and D. S. Trent. 1984. Pressurized Thermal Shock: TEMPEST Computer Code Simulation of Thermal Mixing in the Cold Leg and Downcomer of a Pressurized Water Reactor. NUREG/CR-3564, PNL-4909, prepared for the U.S. Nuclear Regulatory Commission by Pacific Northwest Laboratory, Richland, Washington.

JANAF. 1986. JANAF Thermochemical Tables, 3rd Edition, American Chemical Society and American Institute of Physics. (Journal of Physical and Chemical Reference Data Volume 1485, Supplement No. 1.)

Roache, P. J. 1976. Computational Fluid Dynamics. Hermosa Publishers, Albuquerque, New Mexico.

Trent, D. S., L. L. Eyler. 1985. TEMPEST - Code Simulations of Hydrogen Distribution in Reactor Containment Structures. EPRI NP-3955, prepared for the Electric Power Research Institute by Battelle Pacific Northwest Laboratory, Richland, Washington.

# APPENDIX A

## IMPLICIT VISCOUS DIFFUSION CHANGES TO TEMPEST

# APPENDIX A

## IMPLICIT VISCOUS DIFFUSION CHANGES TO TEMPEST

This appendix contains the HISTORIAN input change file that creates the N29B1 version of the code from the N29A0 version of TEMPEST. The file is provided to maintain traceability and quality assurance on the N29 versions of TEMPEST when applied to canister simulation.

```
*/      name = c_tempst_nb0.upd
*/      path = /800320/tempest/n29/a0
*/
*/      compatible with c_tempst_na0.xxx
*/           which is stored on the path: /800320/tempest/n29/a0
*/
*/      note that these updates create the "b1" version of tempest_n29
*/           from the "a0" version. there is no "b0" version as parent.
*/
*/
*/      ctss updates for the b1 version of tempest_n29
*/           these updates allow for debug output on file outd = tape14
*/           and the use of multiple titles
*/           these updates also create a new card in card group 1 called
*/           the dbug card. on this card the array idbug is loaded.later in
*/           the code it can be used for certain debug operations like debug
*/           writes to output or outd.
*/
*/      starting with line 98 (*id impcomm) are the update
*/           that l.l.eyler put together to form a suitable n29 version
*/           from the base n2 version that we have had floating around.
*/           these changes give the n29 version implicit momentum
*/           diffusion plus some vectorization.
*/
*id dbfile0
*d tempest.34
     3   tape13,tape20,tape21,monsav,tape3=monsav,outd,tape14=outd,
*d vtype.2
      integer out,page,timunt,units,cname,outd,wrdsze,cf1,cf3
*i ncom.35
      common/ex7/outd
*i tempest.293
      outd     = 14
*i input.25
      outd = 14
*id lablfix
*i tempest.325
      ionce = 0
   38 continue
*i tempest.326
      if(ionce.ne.0) go to 42
*i tempest.329
   42 continue
*i tempest.328
      ionce = 1
*i tempest.345
      if(tlabel(20).eq.4h****) go to 38
*d tempest.331
*d tempest.338,345
*id dbswtch0
*i ncom.35
      common/ex8/idbug(200)
```

```
*id dbswtch1
*i tempest.362
      if(xcard.eq.4hdbug) go to 170
*i tempest.497
  170 continue
c**
c**   dbug array card
c**
      igroup = int(data(14))
      if(igroup.le.0) igroup = 1
      if(igroup.gt.20) igroup = 20
      istart = 10*(igroup-1) + 1
      l = istart - 1
      do 175 n = 1,10
      l = l+1
      idbug(l) = int(data(n))
  175 continue
      go to 100
*i plabak.51
      write(out,90000)
      n1 = -49
      i1 = -4
      i2 = 0
      do 2 nn = 1,4
      i1 = i1+5
      i2 = i2+5
      write(out,90003) i1,i2
      n1 = n1+50
      do 1 ll = 1,10
      i = n1+ll-1
      j = i+10
      k = i+20
      l = i+30
      m = i+40
      write(out,90001) i,idbug(i),j,idbug(j),k,idbug(k),l,idbug(l),
     $ m,idbug(m)
    1 continue
    2 continue
      write(out,90002)
90000 format(///,5x,"idbug array settings",///)
90001 format(2x,2hi=,i5,2x,6hidbug=,i5,2x,2hj=,i5,2x,6hidbug=,i5,2x,
     1 2hk=,i5,2x,6hidbug=,i5,2x,2hl=,i5,2x,6hidbug=,i5,2x,2hm=,i5,2x,
     2 6hidbug=,i5)
90002 format(///)
90003 format(//,2x,"idbug array for groups",2x,i3," to ",i3,//)
*id impcomm
*/    vmc array needed for moment implicit
*d comlst.13
      common/imp/vmc(maxarr),smr(maximp),smz(maximp),smx(maximp),
*id impfact
*/      redimension
*d facts.2,16
```

```
      parameter( maxsze = 2500)
      parameter( lsize  = 102)
      parameter( maxsc  = 2600)
      parameter( maxtxt = 4000)
      parameter( maxset =  2000)
c**
      parameter( maxarr = 3*maxsze)
      parameter( maxcnt = 1*maxsze)
      parameter( maximp = maxsze)
      parameter( maxemf = 1 )
      parameter( maxcmp = 1)
      parameter( maxcct = 1)
c**
      parameter( npft   = 100)
*id nfix29
*/
*/      update changes to fix version 29
*/      tempest, cray version n, mod 29.1 on 13jul86.
*/
*/ **************************************************** verson ***********
*/
*/      update current version and mod number
*i tempest.44
c**        correction run    - mod 29.1 -     july 13,   1986         **
*d verson.2,3
      data machin,verson,(datver(j),j=1,2),modnum,(datmod(i),i=1,2)
     1           /"cray","npsl","jul ","1985","29.1","july","1986"/
*/ ****************************************************** common *********
*/
*/      generic common block changes
*/
*/ ***********************************************************************
*/
*/   problem number: n29.#
*/
*/     problem:
*/      finder:  Loren Eyler (pnl)  00mon86
*/       fixer:  Loren Eyler
*/
*/ ***********************************************************************
*/
*/   problem number: n29.1
*/
*/     problem:  tabulated props not working correctly due to if statement
*/      finder:  Loren Eyler (pnl)  13jul86
*/       fixer:  Loren Eyler
*/
*d tabset.259
      if((k/itab)*itabpl.ge.ktab(1)) go to 480
*/  *******************************************************  adindx   **
*/     correct transfer around coding in adindx
*d adindx.85
```

```
      if(.not.(contrl(2).or.contrl(54))) go to 600
*/  ******************************************************* getset   **
*/   do calculation of electrode pair volumes for equating source=sink
*d getset.155,168
cc    if(nef.eq.0) go to 650
cc    do 640 npair = 1,nef
cc    nsave    = epvol(1,npair)
cc    do 630 j = 1,2
cc    nsink    = epvol(j,npair)
cc    eptvol   = 0.
cc    if(nsrc.eq.0) go to 625
cc    do 620 i = ns1,ns2,nsinc
cc    n        = index(i)
cc    ntab     = index(i+4)
cc    if(ntab.eq.nsink) eptvol = eptvol+cv(n)
cc620 continue
cc625 continue
cc    epvol(j,npair) = eptvol
cc630 continue
cc    nepvol(npair) = nsave
cc640 continue
cc650 continue
*id momx
*/
*/      implicit momentum formulation: used by crowder in early melter work
*/     to get to word in tempest, vers n, mod 29.0, all that was changed
*/     was to multiply the fdc's at end of moment by appropriate sc's
*/     and to move call to viscos up front
*/
*/
*d moment.10
      if(contrl(5).or.contrl(15).and..not.skipon) call viscos(nvis)
*d moment.557
*d moment.12,549
      sent     = 1.0
      dtdt     = damp*dt
      summon   = smallr
c**
      do  20 n = 1,jks
      emu(n)   = cvmgt(smallr,emu(n),nt(n).eq.20)
      et(n)    = cvmgt(smallr,et(n),nt(n).eq.20)
   20 continue
      if(skan.and.scribe(9)) call output(4)
      do  30 n = 1,jks
      pk(n)    = 0.
      cf(n)    = 0.
      smr(n)   = 0.
      smz(n)   = 0.
      smx(n)   = 0.
      usr(n)   = 0.
      vsr(n)   = 0.
      wsr(n)   = 0.
      beta(n)  = smallr
      e(n)     = bigger
   30 continue
c**
      do  35 n = 1,mqs
      qs(n)    = smallr
      vmc(n)   = smallr
      fdc(n)   = smallr
   35 continue
c**
c**
      grj      = 0.
      gzk      = gk
      gxi      = 0.
      uic      = 0.
      wjc      = 0.
      if(contrl(4).and..not.contrl(31)) gzk = .0
      if(airdg.le.small) go to 70
      wind     = 1..
      sangle   = wangl
      if(natab.gt.0) sangle = qfac(natab)
      if(nwtab.gt.0) wind  =  qfac(nwtab)
      rwind    = airdg*wind**wexp*cos(sangle)
      xwind    = airdg*wind**wexp*sin(sangle)
   70 continue
c**   temporarily convert west,south and near side
c**   type 40 boundary cells to type 0
      if(next.eq.0) go to 80
      n1       = lset(16)+1
      n2       = lset(19)
      do  75 n = n1,n2,4
      jk       = index(n)
      if(index(n+3).gt.0) go to 75
      nt(jk)   = 0
   75 continue
   80 continue
c**
c**   compute momentum connectors (for half-cell width)
c**
      do 100 i = 1,nix
      ik       = (i-1)*ljk
      ii       = i+kx
      do 100 k = 1,nkz
      kj       = (k-1)*nr+ik
      kk       = k+nr
cdir$ ivdep
      do 100 j = 1,njr
      n        = kj+j
      sr17     = sc(j*lr+j17)
      qr       = 1.
      qz       = 1.
      qx       = 1.
      en        = et(n)+emu(n)
      pk(n)     = en
      ejp1      = et(n+lwn)+emu(n+lwn)
      ekp1      = et(n+lnr)+emu(n+lnr)
      eip1      = et(n+ljk)+emu(n+ljk)
      denomr    = dx(j) /(en*ar(n))+dx(j+1) /(ejp1*ar(n+lwn))
      denomz    = dx(kk)/(en*az(n))+dx(kk+1)/(ekp1*az(n+lnr))
      denomx    = dx(ii)/(en*ax(n))+dx(ii+1)/(eip1*ax(n+ljk))
      vmc(n)    = tdr*qr*dx(kk)*dx(ii)*r(j)/denomr+smallr
      vmc(n+kq)= tdz*qz*dx(ii)*dx(j)*rc(j)/denomz+smallr
      vmc(n+iq)= tdx*qx*dx(kk)*dx(j)* sr17/denomx+smallr
  100 continue
c**
c**
c**
c**   compute momentum drag terms
c**
      do 250 i = 1,nix
      ik        = (i-1)*ljk
      is        = (i-1)*lx+ksc
      sx1       = sc(is+1)
      sx2       = sc(is+2)
      sx9       = sc(is+9)
      do 250 k = 1,nkz
      kj        = (k-1)*nr+ik
      ks        = (k-1)*lz+jsc
      sz1       = sc(ks+1)
      sz2       = sc(ks+2)
      sz9       = sc(ks+9)
      do 250 j = 1,njr
      n         = kj+j
      mat       = mt(n)
      beta(n)   = prop(2,mat)
      ndcn      = ndc(n)
      if(ndcn.eq.0) go to 250
      sr1       = sc(j*lr+j01)
      sr2       = sc(j*lr+j02)
      sr9       = sc(j*lr+j09)
      sr17      = sc(j*lr+j17)
      sr18      = sc(j*lr+j18)
      ncr       = ndcn/npack2
      ncz       = (ndcn-ncr*npack2)/npack
      ncx       = ndcn-ncr*npack2-ncz*npack
      smrr      = 0.
      smzz      = 0.
      smxx      = 0.
      ll        = prop(22,mat)
      if(ll.eq.0) go to 190
      cmu25     = tcon(12,ll)
      tkep      = tke(n)
      frz       = 0.0
      frx       = 0.0
      fzr      = 0.0
      fzx      = 0.0
      fxr      = 0.0
      fxz      = 0.0
      enup     = prop(4,mat)*prop(13,mat)
c**
c**   flow boundary condition
c**
c**   r-direction log-law
      srtke    = sqrt(sr2*tke(n+lwn)+sr1*tkep)*cmu25
      if(ncz.le.99) go to 160
      yplus    = srtke*dx(k+nr)/(2.*enup)
      if(yplus.lt.11.) go to 160
      frz      = cappa*srtke*sr9/alog(elog*yplus)*(ncz/100)
  160 continue
      if(ncx.le.99) go to 165
      yplus    = srtke*dx(i+kx)/(2.*enup*sr18)
      if(yplus.lt.11.) go to 165
      frx      = cappa*srtke*sr9/alog(elog*yplus)*(ncx/100)
  165 continue
      smrr     = frz+frx
c**   z-direction log-law
      srtke    = sqrt(sz2*tke(n+lnr)+sz1*tkep)*cmu25
      if(ncr.le.99) go to 175
      yplus    = srtke*dx(j)/(2.*enup)
      if(yplus.lt.11.) go to 175
      fzr      = cappa*srtke*sz9/alog(elog*yplus)*(ncr/100)
  175 continue
      if(ncx.le.99) go to 178
      yplus    = srtke*dx(i+kx)/(2.*enup*sr17)
      if(yplus.lt.11.) go to 178
      fzx      = cappa*srtke*sz9/alog(elog*yplus)*(ncx/100)
  178 continue
      smzz     = fzr+fzx
c**   x-direction log-law
      srtke    = sqrt(sx2*tke(n+ljk)+sx1*tkep)*cmu25
      if(ncr.le.99) go to 187
      yplus    = srtke*dx(j)/(2.*enup)
      if(yplus.lt.11.) go to 187
      fxr      = cappa*srtke*sx9/alog(elog*yplus)*(ncr/100)
c**   special treatment of x direction flow in cylindrical coords
      if(.not.contrl(1)) go to 187
      scmr     = 0.
      if(nt(n-lwn).gt.49) scmr = sc(j*lr-j18)
      if(nt(n+lwn).gt.49) scmr = scmr+sr18
      fxr      = fxr*scmr
  187 continue
      if(ncz.le.99) go to 188
      yplus    = srtke*dx(k+nr)/(2.*enup)
      if(yplus.lt.11.) go to 188
      fxz      = cappa*srtke*sr17*sx9/alog(elog*yplus)*(ncz/100)
  188 continue
```

```
      smxx     = fxr+fxz
  190 continue
      if(ncr.eq.0.or.ncr.ge.99) go to 192
      aup      = abs(u(n))
      factor   = 0.5*hdc(ncr)
      if(ncr.lt.49)factor= factor*sr9
      if(dex(ncr).ne.2.) factor = factor*(aup+smallr)**(dex(ncr)-2.)
      smrr     = smrr+factor*aup
  192 continue
      if(ncz.eq.0.or.ncz.ge.99) go to 195
      avp      = abs(v(n))
      factor   = 0.5*hdc(ncz)
      if(ncz.lt.49)factor= factor*sz9
      if(dex(ncz).ne.2.) factor = factor*(avp+smallr)**(dex(ncz)-2.)
      smzz     = smzz+factor*avp
  195 continue
      if(ncx.eq.0.or.ncx.ge.99) go to 210
      awp      = abs(w(n))
      factor   = 0.5*hdc(ncx)
      if(ncx.lt.49)factor= factor*sx9*sr17
      if(dex(ncx).ne.2.) factor = factor*(awp+smallr)**(dex(ncx)-2.)
      smxx     = smxx+factor*awp
  210 continue
      smr(n)   = smr(n)+smrr
      smz(n)   = smz(n)+smzz
      smx(n)   = smx(n)+smxx
  250 continue
c**
c**   compute momentum source terms
c**
      do 300 i = nii,nix
      ik       = (i-1)*ljk
      is       = (i-1)*lx+ksc
      sx1      = sc(is+1)
      sx2      = sc(is+2)
      sx9      = sc(is+9)
      sx16     = sc(is+16)*.5
      gj       = gr*sc(is+19)
      gi       = gx*sc(is+20)
      do 300 k = nkk,nkz
      kj       = (k-1)*nr+ik
      ks       = (k-1)*lz +jsc
      sz1      = sc(ks+1)
      sz2      = sc(ks+2)
      sz9      = sc(ks+9)
      sz16     = sc(ks+16)*.5
      szvv     = .25*sz9
      do 300 j = njj,njr
      n        = kj +j
      aq       = cvmgt(1.,0.,nt(n).lt.20)
      sr1      = sc(lr*j+j01)
      sr2      = sc(lr*j+j02)
      sr9       = sc(lr*j+j09)
      sr16      = sc(lr*j+j16)*.5
      sr17      = sc(lr*j+j17)
      sr18      = sc(lr*j+j18)
      sr19      = sc(lr*j+j19)
      sxwv      = sr17*sx16
      sxwu      = sr18*sx16
      sruu      = .5*sr9*sr18
      sxww      = .25*sx9*sr17
      dp        = rho(n)
      dp0       = beta(n)
      denr      = 1./dp0
      sr67      = sr16*sr17
      volr      = .5*(cv(n)+cv(n+lwn))
      volz      = .5*(cv(n)+cv(n+lnr))
      volx      = .5*(cv(n)+cv(n+ljk))
c**
c**   set velocities
c**
      up        = u(n)
      ujp1      = u(n+lwn)
      ujm1      = u(n-lwn)
      uip1      = u(n+ljk)
      uim1      = u(n-ljk)
      ukp1      = u(n+lnr)
      ukm1      = u(n-lnr)
      ujmip     = u(n+jmip)
      ujmkp     = u(n+jmkp)
      vp        = v(n)
      vjp1      = v(n+lwn)
      vjm1      = v(n-lwn)
      vkp1      = v(n+lnr)
      vkm1      = v(n-lnr)
      vip1      = v(n+ljk)
      vim1      = v(n-ljk)
      vjpkm     = v(n+jpkm)
      vkmip     = v(n+kmip)
      wp        = w(n)
      wjm1      = w(n-lwn)
      wjp1      = w(n+lwn)
      wim1      = w(n-ljk)
      wip1      = w(n+ljk)
      wkm1      = w(n-lnr)
      wkp1      = w(n+lnr)
      wjpim     = w(n+jpim)
      wkpim     = w(n+kpim)
c**
c**   compute velocity averages
c**
      uap       = .5*(up+ujp1)
      uam       = .5*(up+ujm1)
      ukc       = .5*(sz2*(ukp1+ujmkp)+sz1*(up+ujm1))
      uic      = .5*(sx2*(uip1+ujmip)+sx1*(up+ujm1))
      vjc      = .5*(sr2*(vjp1+vjpkm)+sr1*(vp+vkm1))
      vic      = .5*(sx2*(vip1+vkmip)+sx1*(vp+vkm1))
      wjc      = .5*(sr2*(wjp1+wjpim)+sr1*(wp+wim1))
      wkc      = .5*(sz2*(wkp1+wkpim)+sz1*(wp+wim1))
c**
c**   compute density averages
c**
      rhojp1   = cvmgt(dp,rho(n+lwn),nt(n+lwn).eq.20.or.nt(n+lwn).gt.49)
      rhokp1   = cvmgt(dp,rho(n+lnr),nt(n+lnr).eq.20.or.nt(n+lnr).gt.49)
      rhoip1   = cvmgt(dp,rho(n+ljk),nt(n+ljk).eq.20.or.nt(n+ljk).gt.49)
      rhor     = cvmgt(sr1*dp+sr2*rhojp1,dp0,couple)
      rhoz     = cvmgt(sz1*dp+sz2*rhokp1,dp0,couple)
      rhox     = cvmgt(sx1*dp+sx2*rhoip1,dp0,couple)
      smr(n)   = rhor*smr(n)
      smz(n)   = rhoz*smz(n)
      smx(n)   = rhox*smx(n)
      usr(n)   = -smr(n)*up
      vsr(n)   = -smz(n)*vp
      wsr(n)   = -smx(n)*wp
c**
c**   compute advective momentum
c**
      duu      = sruu*(rc(j+1)*(abs(uap)*(up-ujp1)*sent +2.*uap*uap)
     1           +     rc(j)  *(abs(uam)*(up-ujm1)*sent -2.*uam*uam))
      dwu      = sxwu*(abs(sr2*wjp1 +sr1*wp)  *(up-uip1)*sent
     1           +     abs(sr2*wjpim+sr1*wim1)*(up-uim1)*sent
     2           +        (sr2*wjp1 +sr1*wp)  *(up+uip1)
     3           -        (sr2*wjpim+sr1*wim1)*(up+uim1))
      dvu      = sz16*(abs(sr2*vjp1 +sr1*vp)  *(up-ukp1)*sent
     1           +     abs(sr2*vjpkm+sr1*vkm1)*(up-ukm1)*sent
     2           +        (sr2*vjp1 +sr1*vp)  *(up+ukp1)
     3           -        (sr2*vjpkm+sr1*vkm1)*(up+ukm1))
      dww      = sxww*(abs(wip1+wp)*(wp-wip1)*sent +wip1*wip1
     1           +     abs(wim1+wp)*(wp-wim1)*sent -wim1*wim1
     2           +     2.*(wip1-wim1)*wp)
      duw      = sr67*(r(j)  *(abs(sx2*uip1 +sx1*up)  *(wp-wjp1)*sent
     1           +     r(j-1)*(abs(sx2*ujmip+sx1*ujm1)*(wp-wjm1)*sent
     2           +              (sx2*uip1 +sx1*up)  *(wp+wjp1))
     3           -              (sx2*ujmip+sx1*ujm1)*(wp+wjm1)))
      dvw      = sz16*(abs(sx2*vip1 +sx1*vp)  *(wp-wkp1)*sent
     1           +     abs(sx2*vkmip+sx1*vkm1)*(wp-wkm1)*sent
     2           +        (sx2*vip1 +sx1*vp)  *(wp+wkp1)
     3           -        (sx2*vkmip+sx1*vkm1)*(wp+wkm1))
      dvv      = szvv*(abs(vkp1+vp)*(vp-vkp1)*sent +vkp1*vkp1
     1           +     abs(vkm1+vp)*(vp-vkm1)*sent -vkm1*vkm1
     2           +     2.*(vkp1-vkm1)*vp)
      duv      = sr67*(r(j)  *(abs(sz2*ukp1 +sz1*up)  *(vp-vjp1)*sent
     1           +     r(j-1)*(abs(sz2*ujmkp+sz1*ujm1)*(vp-vjm1)*sent
     2           +              (sz2*ukp1 +sz1*up)  *(vp+vjp1))
     3           -              (sz2*ujmkp+sz1*ujm1)*(vp+vjm1)))
      dwv      = sxwv*(abs(sz2*wkp1 +sz1*wp)  *(vp-vip1)*sent
     1          +     abs(sz2*wkpim+sz1*wim1)*(vp-vim1)*sent
     2          +        (sz2*wkp1 +sz1*wp)  *(vp+vip1)
     3          -        (sz2*wkpim+sz1*wim1)*(vp+vim1))
c**
      usorc    = (duu +dwu +dvu)*dp0
      wsorc    = (duw +dww +dvw)*dp0
      vsorc    = (duv +dwv +dvv)*dp0
c**
c**   compute cylindrical centripetal and coriolis terms
c**
      centrp   = -cyl*sr18*wjc*wjc
      coriol   =  cyl*sr17*uic
      usorc    = usorc-centrp*rhor
      smx(n)   = smx(n)+coriol*rhox
c**
c**   compute coriolis contribution for large scale geophysical flows
c**
      phiu     = sc(j*lr+j25)+sc(is+22)
      phiw     = sc(j*lr+j26)+sc(is+23)
      phiv     = sc(j*lr+j26)+sc(is+22)
      usorc    = usorc -eomeg*rhor*(sin(phiu)*wjc-cos(phiu)*vjc)
      wsorc    = wsorc +eomeg*rhox*sin(phiw)*uic
      vsorc    = vsorc -eomeg*rhoz*cos(phiv)*ukc
c**
c**   compute gravitational forcing functions
c**
      usorc    = usorc-cvmgt((rhor-baxp*dp0)*gj,0.,couple)
      vsorc    = vsorc-cvmgt((rhoz-baxp*dp0)*gk,0.,couple)
      wsorc    = wsorc-cvmgt((rhox-baxp*dp0)*gi,0.,couple)
c**
c**   complete formulation of momentum sources
c**
      usr(n)   = aq*(usr(n)-usorc)*volr
      vsr(n)   = aq*(vsr(n)-vsorc)*volz
      wsr(n)   = aq*(wsr(n)-wsorc)*volx
      smr(n)   = smr(n)*denr
      smz(n)   = smz(n)*denr
      smx(n)   = smx(n)*denr
c**
c**   compute courant stability
      cf(n)    = abs(up)*sr9+abs(vp)*sz9+abs(wp)*sx9*sr17
  300 continue
c**
c**   check courant stability
c**
      ism      = isamax(jks,cf,1)
      cfmax    = cf(ism)
      summon   = amax1(cfmax,summon)
      if(nstep.eq.0) go to 805
      if(summon*dt.gt.1.) bakup = .true.
      if(bakup) go to 800
      do 310 n = 1,jks
      e(n)     = 0.
  310 continue
c**
      if(nr.le.3) go to 450
c**   compute new time explicit u-velocity
c**
      do 400 i = 1,nix
      ik       = (i-1)*ljk
      ii       = i+kx
      do 400 k = 1,nkz
      kj       = (k-1)*nr+ik
      kk       = k+nr
cdir$ ivdep
      do 400 j = 1,njr
      n        = kj+j
      sr16     = sc(j*lr+j16)
      sr18     = sc(j*lr+j18)
      sr20     = sc(lr*j+j20)
      sr21     = sc(lr*j+j21)
      srp      = sc(j*lr+j09)*gset
      rq       = cvmgt(1.,smallr,nt(n).lt.50.and.nt(n).ne.20)
      usr(n)   = usr(n)-srp*rq*(p(n+lwn)-p(n))*.5*(cv(n)+cv(n+lwn))
      qs(n)    = pk(n)*ar(n)*dx(kk)*dx(ii)*sr18*sr16
      qs(n+kq) = .5*(1.+sr21)*vmc(n+kq)+.5*(1.+sr20)*vmc(n+kqjp)
      qs(n+iq) = vmc(n+iq)+vmc(n+iqjp)
      fdc(n)   = qs(n)
      fdc(n+kq)= qs(n+kq)
      fdc(n+iq)= qs(n+iq)
  400 continue
      do 420 n = npst,nped
      dp0      = beta(n)
      vq       = cvmgt(1.,0.,nt(n).lt.20)
      tims     = 2./(dp0*(cv(n)+cv(n+lwn)))/(1.+smr(n)*dt)
      sumtot   = qs(n+lwn)+fdc(n)  + qs(n+kq)+fdc(n+kqm)+
     1           qs(n+iq) +fdc(n+iqm)
      e(n)     =(qs(n)    *u(n-lwn)+ fdc(n+lwn)*u(n+lwn)+
     1           qs(n+kqm)*u(n-lnr)+ fdc(n+kq) *u(n+lnr)+
     2           qs(n+iqm)*u(n-ljk)+ fdc(n+iq) *u(n+ljk)-
     3           sumtot*u(n) +usr(n))*tims*dt*vq
      beta(n)  = tims*frnk*dt*vq
      usr(n)   = vq*sumtot
      smr(n)   = smr(n)+frnk*vq*sumtot/dp0
  420 continue
c**
c**   u-velocity implicit continuation
c**
      if(lset(7).eq.lset(8)) go to 433
      n1       = lset(7)+1
      n2       = lset(8)
      do 432 n = n1,n2
      jk       = index(n)
      e(jk)    = 0.
  432 continue
  433 continue
      call solvex(25,25,27,jks,e)
  435 continue
c**
c**   update u-velocities
c**
      do 440 n = 1,jks
      u(n)     = u(n)+cvmgt(e(n),0.,nt(n).lt.20)
  440 continue
  450 if(nz.le.3) go to 550
      do 460 n = 1,jks
      mat      = mt(n)
      beta(n)  = prop(2,mat)
  460 continue
c**
c**
c**   compute new time explicit v-velocity
c**
      do 500 i = 1,nix
      ik       = (i-1)*ljk
      ii       = i+kx
      do 500 k = 1,nkz
      kj       = (k-1)*nr+ik
      kk       = k+nr
      ks       = (k-1)*lz+jsc
      sz9      = sc(ks+9)
      sz16     = sc(ks+16)
cdir$ ivdep
      do 500 j = 1,njr
      n        = kj+j
      sr17     = sc(j*lr+j17)
      sr18     = sc(j*lr+j18)
      szp      = sz9*gset
      rq       = cvmgt(1.,smallr,nt(n).lt.50.and.nt(n).ne.20)
      vsr(n)   = vsr(n)-szp*rq*(p(n+lnr)-p(n))*.5*(cv(n)+cv(n+lnr))
      qs(n)    = vmc(n)+vmc(n+lnr)
      qs(n+kq) = pk(n)*az(n)*dx(j)*dx(ii)*sr17*sz16
      qs(n+iq) = vmc(n+iq) +vmc(n+iqkp)
      fdc(n)   = qs(n)
      fdc(n+kq)= qs(n+kq)
      fdc(n+iq)= qs(n+iq)
  500 continue
      do 520 n = npst,nped
      dp0      = beta(n)
      vq       = cvmgt(1.,0.,nt(n).lt.20)
      tims     = 2./(dp0*(cv(n)+cv(n+lnr)))/(1.+smz(n)*dt)
      sumtot   = qs(n)+fdc(n-lwn)+ qs(n+kqp)+fdc(n+kq)+
     1           qs(n+iq) +fdc(n+iqm)
      e(n)     =(qs(n-lwn)*v(n-lwn)+ fdc(n)    *v(n+lwn)+
     1           qs(n+kq) *v(n-lnr)+ fdc(n+kqp)*v(n+lnr)+
     2           qs(n+iqm)*v(n-ljk)+ fdc(n+iq) *v(n+ljk)-
     3           sumtot*v(n) +vsr(n))*tims*dt*vq
      beta(n)  = tims*frnk*dt*vq
      vsr(n)   = vq*sumtot
      smz(n)   = smz(n)+frnk*vq*sumtot/dp0
  520 continue

c**
c**   v-velocity implicit continuation
c**
      if(lset(8).eq.lset(9)) go to 533
      n1       = lset(8)+1
      n2       = lset(9)
      do 532 n = n1,n2
      jk       = index(n)
      e(jk)    = 0.
  532 continue
  533 continue
c**
      call solvex(26,25,27,jks,e)
  535 continue
c**
c**   update v-velocities
c**
      do 540 n = 1,jks
      v(n)     = v(n)+cvmgt(e(n),0.,nt(n).lt.20)
  540 continue
  550 if(nx.le.3) go to 650
      do 560 n = 1,jks
      mat      = mt(n)
      beta(n)  = prop(2,mat)
  560 continue
c**
c**
c**   compute new time explicit w-velocity
c**
      do 600 i = 1,nix
      is       = (i-1)*lx+isc
      ik       = (i-1)*ljk
      ii       = i+kx
      sx9      = sc(is+9)
      sx16     = sc(is+16)
      do 600 k = 1,nkz
      kj       = (k-1)*nr+ik
      kk       = k+nr
cdir$ ivdep
      do 600 j = 1,njr
      n        = kj+j
      sr17     = sc(j*lr+j17)
      sxp      = sx9*sr17*gset
      rq       = cvmgt(1.,smallr,nt(n).lt.50.and.nt(n).ne.20)
      wsr(n)   = wsr(n)-sxp*rq*(p(n+ljk)-p(n))*.5*(cv(n)+cv(n+ljk))
      qsn      = vmc(n)+vmc(n+ljk)
      qs(n)    = qsn*sr17
      qs(n+kq) = vmc(n+kq) +vmc(n+kqip)
      qs(n+iq) = pk(n)*ax(n)*dx(kk)*dx(j)*sr17*sx16
      fdc(n)   = qsn/rc(j+1)
      fdc(n+kq)= qs(n+kq)
      fdc(n+iq)= qs(n+iq)
  600 continue
      do 620 n = npst,nped
      dp0      = beta(n)
      vq       = cvmgt(1.,0.,nt(n).lt.20)
      tims     = 2./(dp0*(cv(n)+cv(n+ljk)))/(1.+smx(n)*dt)
      sumtot   = qs(n)+fdc(n-lwn)+ qs(n+kq)+fdc(n+kqm)+
     1           qs(n+iqp) +fdc(n+iq)
      e(n)     =(qs(n-lwn)*w(n-lwn)+ fdc(n)    *w(n+lwn)+
     1           qs(n+kqm)*w(n-lnr)+ fdc(n+kq) *w(n+lnr)+
     2           qs(n+iq) *w(n-ljk)+ fdc(n+iqp)*w(n+ljk)-
     3           sumtot*w(n) +wsr(n))*tims*dt*vq
      beta(n)  = tims*frnk*dt*vq
      wsr(n)   = vq*sumtot
      smx(n)   = smx(n)+frnk*vq*sumtot/dp0
  620 continue
c**
c**   w-velocity implicit continuation
c**
      if(lset(9).eq.lset(10)) go to 633
      n1       = lset(9)+1
      n2       = lset(10)
      do 632 n = n1,n2
      jk       = index(n)
      e(jk)    = 0.
  632 continue
  633 continue
c**
      call solvex(27,25,27,jks,e)
  635 continue
c**
c**   update w-velocities
c**
      do 640 n = 1,jks
      w(n)     = w(n)+cvmgt(e(n),0.,nt(n).lt.20)
  640 continue
  650 continue
c**
      if(nstep.eq.0) go to 805
c**
c**   set fdc array for pressure iteration
c**
cdir$ ivdep
*/    following loop was only change to get to work in vers. n, mod 29.0
      do 700 i = 1,nx
      ik       = (i-1)*ljk
      is       = (i-1)*lx+ksc
      do 700 k = 1,nz
      kj       = (k-1)*nr+ik
      ks       = (k-1)*lz+jsc
      do 700 j = 1,nr
      n        = j+kj
      js       = (j-1)*lr
      pq       = cvmgt(1.,smallr,nt(n).lt.20)
      fdc(n)   = pq*tdr/(1.+smr(n)*dt)*sc(js+9)
      fdc(n+kq)= pq*tdz/(1.+smz(n)*dt)*sc(ks+9)
      fdc(n+iq)= pq*tdx/(1.+smx(n)*dt)*sc(is+9)*sc(js+17)
  700 continue
c**
      if(nsrc.gt.0) call mosrc(1)
  800 continue
c**
c**   reestablish type 40 cell that were converted to type 0
c**
  805 if(next.eq.0) go to 820
      n1       = lset(16)+1
      n2       = lset(19)
      do 810 n = n1,n2,4
      jk       = index(n)
      if(nt(jk).eq.0) nt(jk) = 40
  810 continue
  820 if(nstep.eq.0) return
c**
*id bousetx
*d bouset.91,92
      v(jmkm)  = 0.
      w(jmkm)  = 0.
*d bouset.101,102
      u(jmkm)  = 0.
      w(jmkm)  = 0.
*d bouset.111,112
      u(jmkm)  = 0.
      v(jmkm)  = 0.
*deck solvex
      subroutine solvex(mode,ll1,ll2,jkss,fsl)
c**
c**   vectorized tridiagonal matrix gauss elimination with triple sweep
c**     (solvex is solvet except for use of lnat to shift qs and fdc)
c**   transient parabolic implicit equation solver
c**
      dimension fsl(jkss)
c**
      logical    cyclic
*call facts
*call comlst
*call ncom
*call vtype
c**
c**
```

```
c**   implicit time step continuation
c**
      do 190 ll = ll1,ll2
      lode     = cvmgt(1,0,ll.eq.mode)
      l1       = lset(ll-1)+1
      l2       = lset(ll)
      if(l1-1.eq.l2) go to 190
      nat      = index(l1)
      nm       = index(l1+1)
      l1       = l1+2
      do 185 l = l1,l2,2
      n1       = index(l)
      n3       = index(l+1)-lode
      n2       = n1+n3*nat
      n4       = n3+1
      n5       = n2+nat
      knat     = cvmgt(0,nat,ll.eq.mode)
      lnat     = cvmgt(nat,0,ll.eq.mode)
      m        = 0
      cyclic   = .false.
      if(nt(n5).eq.25) cyclic = .true.
      do 150 n = n1,n2,nat
      m        = m+1
      betaf    = beta(n)
      nn       = n+nm
      am(m)    = betaf*qs(nn-knat)
      ap(m)    = betaf*fdc(nn+lnat)
      ac(m)    = (qs(nn+lnat)+fdc(nn-knat))*betaf+1.
      bn(m)    = e(n)
  150 continue
      bet(1)   = 1./ac(1)
      ac(1)    = am(1)*bet(1)
      ap(1)    = ap(1)*bet(1)
      bn(1)    = bn(1)*bet(1)
      apm      = ap(m)
      bnm      = bn(m)
      do 165 k = 2,m
      bet(k)   = 1./(ac(k)-am(k)*ap(k-1))
      ap(k)    = ap(k)*bet(k)
      bn(k)    = (bn(k)+am(k)*bn(k-1))*bet(k)
  165 continue
      if(cyclic) go to 172
      e(n2)    = bn(m)
      do 170 k = 1,n3
      e(n2-k*nat)    = bn(n4-k)+ap(n4-k)*e(n5-k*nat)
  170 continue
      go to 185
  172 continue
c**   cyclic matrix
      amm      = am(m)
      acm      = ac(m)
      m1       = m-1
      m2       = n1-nat
      do 173 k = 2,m1
      ac(k)    = am(k)*ac(k-1)*bet(k)
  173 continue
      ac(m)    = 1.
      am(m)    = 0.
      do 174 lq= 1,m1
      k        = m-lq
      ac(k)    = ap(k)*ac(k+1)+ac(k)
      am(k)    = ap(k)*am(k+1)+bn(k)
  174 continue
      vend     = (bnm+amm*am(m-1)+apm*am(1))/
     1           (acm-amm*ac(m-1)-apm*ac(1))
      do 175 k = 1,m
      e(k*nat+m2)    = ac(k)*vend+am(k)
  175 continue
  185 continue
  190 continue
c**
c**   implicit  continuation complete
c**
      return
      end
```

# APPENDIX B

## FRONT TRACKER CHANGES TO TEMPEST

# APPENDIX B

## FRONT TRACKER CHANGES TO TEMPEST

This appendix contains the HISTORIAN input change file that creates the N29B2 version of TEMPEST from the N29B1 version. These changes are provided to maintain traceability and quality assurance on the N29B2 version of TEMPEST (the latest version with the front tracker model). Note that this version of TEMPEST is not completed. However, this change file will facilitate the restart of the code development project whenever sufficient funding becomes available.

```
*/     name = c_tempst_nbl.uzl  (best front tracker logic 8/1/87 )
*/
*/
*/     name = c_tempst_nbl.upl
*/
*/          stored on path = /800320/tempest/n29/ctss/b1
*/
*/     these updates are debug updates for the n29b1 version of tempest
*/     they allow automatic time stepping based on temperature changes.
*/     they utilize two new logical variables pacet (for temperature) and
*/     pacec (for concentration) .
*/     they also have the mapfix updates in that allow for seeing the
*/       cell types in each of the three coordinate directions.
*/
*id dtlim0
*b vtype.3
      logical pacet,pacec
*i ncom.35
      common/ex4/
     1           dtmaxa,dcmaxa,dtymax,rngetu,rngetd,rngecu,
     2           rngecd,cnge,pacet,pacec,maxtcel,maxccel,dcmax,dti,dst,
     3           maxcel(50),dtmaxx(50)
*id mapfix0
*b ncom.36
      common/ex6/mapo(10,3),xmaplst(3)
*b tempest.285
      data(xmaplst(i),i=1,3)/"rzio","zxio","rxio"/
*d tempest.282
      data(option(i), i=1,15)/"aout","cont","plot","dbug",
     1    "film","mapo",9*"xxxx"/
*id redim
*d impfact.1,14
      parameter( maxsze = 2500)
      parameter( lsize  = 102)
      parameter( maxsc  = 2600)
      parameter( maxtxt = 4000)
      parameter( maxset =  2000)
c**
      parameter( maxarr = 3*maxsze)
      parameter( maxcnt = 1*maxsze)
      parameter( maximp = maxsze)
      parameter( maxemf = 1 )
      parameter( maxcmp = 1)
      parameter( maxcct = 1)
c**
      parameter( npft   = 100)
c**
*/     name = c_tempst_nbl.up2
*/
*/          stored on path = /800320/tempest/n29/ctss/b1
*/
*/     these updates are debug updates for the n29b1 version of tempest
*/     they allow automatic time stepping based on temperature changes.
*/     they utilize two new logical variables pacet (for temperature) and
*/     pacec (for concentration) .
*/     they also have the mapfix updates in that allow for seeing the
*/       cell types in each of the three coordinate directions.
*/
*id mapfix1
*i tempest.647
      if(xlist.eq.option(7)) go to 1500
*d tempest.648
      if(xlist.eq."    ") go to 400
      go to 300
*i tempest.689
 1500 continue
      if(dat(1).eq."ntyp") go to 1510
      if(dat(1).eq."mtyp") go to 1520
      if(dat(1).eq."ptyp") go to 1530
      if(dat(1).eq."ttyp") go to 1540
      if(dat(1).eq."drag") go to 1550
      if(dat(1).eq."film") go to 1560
      go to 1570
 1510 continue
      do 1515 l=2,4
      do 1515 i=1,3
 1515 if(dat(l).eq.xmaplst(i)) mapo(1,i)=1
      go to 100
 1520 continue
      do 1525 l=2,4
      do 1525 i=1,3
 1525 if(dat(l).eq.xmaplst(i)) mapo(2,i)=1
      go to 100
 1530 continue
      do 1535 l=2,4
      do 1535 i=1,3
 1535 if(dat(l).eq.xmaplst(i)) mapo(3,i)=1
      go to 100
 1540 continue
      do 1545 l=2,4
      do 1545 i=1,3
 1545 if(dat(l).eq.xmaplst(i)) mapo(4,i)=1
      go to 100
 1550 continue
      do 1555 l=2,4
      do 1555 i=1,3
 1555 if(dat(l).eq.xmaplst(i)) mapo(5,i)=1
      go to 100
 1560 continue
      do 1565 l=2,4
      do 1565 i=1,3
 1565 if(dat(l).eq.xmaplst(i)) mapo(6,i)=1
      go to 100
 1570 continue
```

```
*i tempest.303
      do 9 i=1,10
      do 9 j=1,3
    9 mapo(i,j)=0
*i tempest.981
      do 590 i = 1,10
      do 590 j = 1,3
      write(out,80001) i,j,mapo(i,j)
80001 format(2x,2hi=,i5,2x,2hj=,i5,2x,5hmapo=,i5)
  590 continue
*d mapper.2
      subroutine mapper(narray,l,label,iplane)
      logical    jkios,kiios,jiios
*d mapper.5
      logical    jkio,jiio,kiio
*i mapper.9
      jkios    = jkio
      jiios    = jiio
      kiios    = kiio
*d mapper.13,15
      if(kiio) nxro = nr
      if(kiio) nrend = nx
      if(iplane.eq.0) go to 8
      if(iplane.eq.1) go to 7
      if(iplane.eq.2) go to 6
c**
c**   rxio planes iplane=3
c**
      jkio     = .false.
      kiio     = .false.
      jiio     = .true.
      go to 200
    6 continue
c**
c**   zxio planes iplane=2
c**
      jkio     = .false.
      kiio     = .true.
      jiio     = .false.
      nrend=nx
      nxro =nr
      go to 9
    7 continue
c**
c**   rzio planes iplane=1
c**
      jkio     = .true.
      kiio     = .false.
      jiio     = .false.
      go to 9
    8 continue
    9 continue
```

```
*d mapper.41
      go to 999
*i mapper.61
  999 continue
      jkio     = jkios
      jiio     = jiios
      kiio     = kiios
*d lcmmap.2
      subroutine lcmmap(mode,narray,l,label,iplane)
      logical    jkios,kiios,jiios
*i lcmmap.10
      jiios    = jiio
      kiios    = kiio
      jkios    = jkio
      if(iplane.eq.0) go to 8
      if(iplane.eq.1) go to 7
      if(iplane.eq.2) go to 6
c**
c**   rxio planes iplane=3
c**
      jiio     = .true.
      kiio     = .false.
      jkio     = .false.
      nrend=nr
      nxz  =nz
      nzx  =nx
      go to 9
    6 continue
c**
c**   zxio planes iplane=2
c**
      jiio     = .false.
      kiio     = .true.
      jkio     = .false.
      nrend=nx
      nxz  =nr
      nzx  =nz
      go to 9
    7 continue
c**
c**   rzio planes iplane=1
c**
      jiio     = .false.
      kiio     = .false.
      jkio     = .true.
      nrend=nr
      nxz  =nx
      nzx  =nz
      go to 9
    8 continue
*i lcmmap.61
      jiio     = jiios
      kiio     = kiios
      jkio     = jkios
*d plabak.350
      isum     = mapo(1,1) + mapo(1,2) + mapo(1,3)
      if(isum.eq.0) call mapper (nt,maxsze,labels(1,1),0)
      if(mapo(1,1).eq.1) call mapper (nt,maxsze,labels(1,1),1)
      if(mapo(1,2).eq.1) call mapper (nt,maxsze,labels(1,1),2)
      if(mapo(1,3).eq.1) call mapper (nt,maxsze,labels(1,1),3)
*d plabak.356
      isum     = mapo(2,1) + mapo(2,2) + mapo(2,3)
      if(isum.eq.0) call lcmmap(1,mt,maxsze,labels(1,2),0)
      if(mapo(2,1).eq.1) call lcmmap(1,mt,maxsze,labels(1,2),1)
      if(mapo(2,2).eq.1) call lcmmap(1,mt,maxsze,labels(1,2),2)
      if(mapo(2,3).eq.1) call lcmmap(1,mt,maxsze,labels(1,2),3)
*d plabak.380
       call lcmmap (2,ndc,maxsze,labels(1,3),0)
*d plabak.381
       call lcmmap (3,ndc,maxsze,labels(1,4),0)
*d plabak.382
       call lcmmap (4,ndc,maxsze,labels(1,5),0)
*d plabak.384
  183 continue
       call lcmmap (2,ndc,maxsze,labels(1,6),0)
*d plabak.385
       call lcmmap (3,ndc,maxsze,labels(1,7),0)
*d plabak.386
       call lcmmap (4,ndc,maxsze,labels(1,8),0)
*d plabak.389
       call lcmmap (2,nhc,maxsze,labels(1,9),0)
*d plabak.390
       call lcmmap (3,nhc,maxsze,labels(1,10),0)
*d plabak.391
       call lcmmap (4,nhc,maxsze,labels(1,11),0)
*id drfix
*i input.505
      do 212 j = 1,nr
      dx(j) = drc*dkind(1,nit)*dkant
  212 continue
      n1 = nr+1
      do 213 k = n1,kx
      dx(k) = dzc*dkind(1,nit)*dkant
  213 continue
      n1 = kx+1
      do 214 i = n1,ix
      dx(i) = dxc*dkind(1,nit)*dkan*cyc
  214 continue
*id tymchek
*i condif.2
      logical timeup
*i condif.11
      timeup   = .false.
*i condif.214
```

```
      if(mod(liters,5).eq.0) call dattim(3,tchek,alph)
      if(tchek+dst.gt.cpulim) timeup = .true.
*i condif.215
      if(timeup) go to 760
*i condif.376
      if(timeup) go to 800
*id dtfix
*b tempest.363
      if(xcard.eq."conc") go to 180
*i setin.161
      pacet    = .false.
      pacec    = .false.
*b tempest.498
  180 continue
  184 continue
c**
c**   qrad card - radiation solution parameters
c**
      read(inrec,10002) acard,(data(i),i=1,14),idat
      iprup    = ifix(data(1))
      go to 100
  186 continue
c**
c**    pack card - bit packing parameters and machine dependencies
c**
      read(inrec,10002) acard,(data(i),i=1,14),idat
      wrdsze   = ifix(data(1))
      ipak     = ifix(data(2))
      isze     = ifix(data(3))
      nlpw     = ifix(data(4))
      go to 100
  188 continue
c**
c**
c**
      go to 100
*i tempest.562
      if(dtymax.eq.0.0) dtymax = big
      dti      = dt
*i tempest.613
      if(dtmaxa.eq.0.0) dtmaxa = 5.0
      if(rngetu.eq.0.0) rngetu = 1.25
      if(rngetd.eq.0.0) rngetd = .75
c**
c**   conc card default values
c**
      if(dcmaxa.eq.0.0) dcmaxa = .01
      if(rngecu.eq.0.0) rngecu = 1.25
      if(rngecd.eq.0.0) rngecd = .75
      cnge     = .90
c**
c**   frez card default values
```

```
c**
c**
c**   mult card default values
c**
c**
c**   qrad card default values
c**
      if(iprup.lt.1) iprup = 10
c**
c**   pack card default values
c**
      if(wrdsze.lt.smallr) wrdsze = 64
      if(ipak.lt.smallr) ipak = 4
      if(nbpl.lt.smallr) nbpl = 15
      if(nlpw.lt.smallr) nlpw = 4
      if(isze.lt.smallr) isze = 100
      nbpl = wrdsze/ipak
*i tempest.1047
10002 format(a4,1x,14f5.0,i5)
*id dtlim1
*/     compatible with lsrcu1 llib1 and cray version a1
*i excute.19
c     dti       = dt + smallr
*d excute.135
*i excute.82
      dtscal    = dtstab*scale
*d excute.221,222
      dtscal    = dtstab*scale
*d excute.229
      if(contrl(30)) dtscal= dtscal*nstep
*i excute.370
      if(mout.and.scribe(8)) call output(3)
*/     updtfix1
*/               compatible with llanlsrcu0 llanllib0
*i excute.19
      dtstab=big
      dtscal=big
      dttlim=big
      dtclim=big
      tblmin=big
      dt6=big
      line=5
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*i excute.124
      if(dti.lt.dt.and.dti.gt.smallr) dt = dti
      line=8
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*d excute.125
      dt        = dti
*i excute.128
```

```
      if(dti.lt.smallr) dti = smallr
      icount   = 0
      dttlim   = big
      dtclim   = big
      dtstab   = big
*i excute.143
      dtstab  = cmax/summax
      line=10
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
 9777 format(/,2x,5hline=,i5,2x,6hnstep=,i5,/,2x,4hdt =,1pe12.5,2x,4hdt1=
     1 ,e12.5,2x,4hdt2=,e12.5,2x,6hdtold=,e12.5,2x,7hdttlim=,e12.5,2x,
     2 7hdtstab=,e12.5,/2x,4hdt3=,e12.5,2x,4hdt4=,e12.5,2x,4hdt5=,
     3 e12.5,2x,6hdtmin=,e12.5,2x,7hdtclim=,e12.5,2x,7htblmin=,e12.5,
     4 /,2x,4hdt6=,e12.5,2x,4hdt7=,e12.5,2x,4hdti=,e12.5)
      dtold    = dt
      dttold   = dt
      dtcold   = dt
      line=20
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*d excute.144,146
      dtstab   = cmax/summax
      line=30
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*i excute.164
      tblmin = big
      if(dtmin.gt.dtymax) dtmin = dtymax
      line=40
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
c     write(out,9778) dtmax,dtmaxa,rngetd,rngetu,cnge,maxtcel
*i excute.218
      line=50
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
      if(.not.pacet) go to 132
      if(abs(dtmax).lt.rngetd*dtmaxa.or.abs(dtmax).gt.dtmaxa*rngetu)
     1       dttlim   = dtold*(dtmaxa/abs(dtmax))*cnge
      line=60
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
c     write(out,9778) dtmax,dtmaxa,rngetd,rngetu,cnge,maxtcel
 9778 format(/,5x,6hdtmax=,1pe12.5,2x,7hdtmaxa=,e12.5,2x,7hrngetd=,
     1 e12.5,2x,7hrngetu=,e12.5,2x,5hcnge=,e12.5,2x,8hmaxtcel=,i5)
  132 continue
      line=70
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
      if(.not.pacec) go to 133
      if(abs(dcmax).lt.rngecd*dcmaxa.or.abs(dcmax).gt.dcmaxa*rngecu)
     1      dtclim   = dtold*(dcmaxa/abs(dcmax))*cnge
      line=80
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
c     write(out,9779) dcmax,dcmaxa,rngecd,rngecu,cnge,maxccel
 9779 format(/,5x,6hdcmax=,1pe12.5,2x,7hdcmaxa=,e12.5,2x,7hrngecd=,
     1 e12.5,2x,7hrngecu=,e12.5,2x,5hcnge=,e12.5,2x,8hmaxccel=,i5)
  133 continue
      line=90
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*d excute.220
      dtstab   = cmax/summax
      line=100
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
*d excute.230
*d excute.240,242
      dt7      = tstop - told
      line=110
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
      dtn    = amin1(dt1,dt2,dt3,dt4,dt5,dt7,dt8,dttlim,dtclim)
      dtmin    = amin1(dtn,dt6,dtstab)
      if(dtmin.le.0.0) go to 810
      if(nstep.eq.nbeg.and.dtmin.gt.dti) dtmin = dti
*/         updtfix
      if(dtmin.gt.2*dtold) dtmin = 2*dtold
      line=120
c     write(out,9777) line,nstep,dt,dt1,dt2,dtold,dttlim,dtstab,
c    1 dt3,dt4,dt5,dtmin,dtclim,tblmin,dt6,dt7,dti
      if(dtmin.gt.2*dtold) dtmin=2*dtold
*i excute.497
      icount   = icount + 1
c     write(out,1022) nstep,icount,dt,dt1,dt2,dt3,dt4,dt5,dttlim,dtclim,
c    1                dtmin,dtstab
      if(icount.gt.5) go to 895
*i excute.525
  895 continue
c     write(out,1028) nstep,icount,dt,dt1,dt2,dt3,dt4,dt5,dttlim,dtclim,
c    1                dtmin,dtstab
*i excute.603
 1028 format(///,2x,"exceeded icount limit in automatic time stepping ro
     1utine in excute. check table values,time step limiters or code lo
     2gic."///,2x,6hnstep=,i5,2x,7hicount=,i5,/,2x,4hdt =,1pe12.5,2x,
     34hdt1=,e12.5,2x,4hdt2=,e12.5,2x,4hdt3=,e12.5,2x,4hdt4=,e12.5,2x,
     44hdt5=,e12.5,//,2x,7hdttlim=,e12.5,2x,7hdtclim=,e12.5,2x,6hdtmin=,
     5e12.5,2x,7hdtstab=,e12.5,///)
*id divgl
*d condif.590
      crashd = .true.
      mode = 3
      if(scribe(13)) call patter
*i patter.214
      if(crashd) stop
*i patter.223
      if(crashd.and.mode.eq.3) call exit
*id dtmont1
*i condif.274
      maxtcel=0
      if(icount.ge.50) icount = 1
      icount = icount + 1
      maxcel(icount) = 0
      dtmaxx(icount) = 0.0
*d condif.280
      if(abs(e(n)).le.abs(dtmax)) go to 177
      dtmax   = e(n)
      maxtcel = n
      dtmaxx(icount) = e(n)
      call locati(n,jcel,kcel,icel)
      maxcel(icount) = icel+100*kcel+10000*jcel
  177 continue
*i condif.307
      kount = kount + 1
*i condif.263
      if(abs(sdepr).le.abs(dtmaxx(kount))) go to 369
      ncel = n2
      dtmaxx(kount) = sdepr
      call locati(ncel,jcel,kcel,icel)
      maxcel(kount) = icel+100*kcel+10000*jcel
  369 continue
*i solvef.70
      if(abs(sdepr).le.abs(dtmaxx(kount))) go to 175
      ncel = n2-k*nat
      call locati(ncel,jcel,kcel,icel)
      maxcel(kount) = icel+100*kcel+10000*jcel
      dtmaxx(kount) = sdepr
*d condif.293
*id bigmon1
*/          compatible with lsrcu3 and llib3
*i input.3
      dimension data1(13),ndata1(13)
*i input.212
      do 101 i=1,10
  101 ndata1(i) = ndata(i)
      ndata1(11) = jm
      ndata1(12) = km
      ndata1(13) = im
*i input.470
      do 201 i = 1,10
  201 data1(i)  = data(i)
      data1(11)= ji
      data1(12)= ki
      data1(13)= ii
```

```
*d input.394,397
*d input.398,401
*d input.404,415
      m = -2
      do 152 n = 1,4
      l = 4*(im-1) + n
      m = m+3
      mon(l) = locate(ndata1(m),ndata1(m+1),ndata1(m+2))
  152 continue
*i input.1286
c**
c**   load mint array for monitor output  (first and second card only)
c**
  850 continue
      m = -2
      do 855 l = 1,8
      n = mon(l)
      call locati(n,j,k,i)
      m = m+3
      mint(m) = j
      mint(m+1) = k
      mint(m+2) = i
  855 continue
*d input.416,419
*id llefix1
*d momx.524
      is       = (i-1)*lx+ksc
*id mltgrd0
*b ncom.36
      common/gridx/ grdindx(maxgindx),grdcord(maxgcord),ngrid,maxgrid,
     1              ispgrd,kfrnt,ncpf
*i facts.17
      parameter(maxgindx = 2000)
      parameter(maxgcord = 2000)
*i tempest.60
      integer    grdtyp
*i setin.75
      maxgrid = 64
      ispgrd = 10
*i setin.294
      ngrid = 0
*/     name = upz1
*/
*/     beginning f-tracker/multi-grid updates
*/
*/
*/     compatible with c_tempst_nb1.xxx
*/
*id acbfx1
*d comlst.12
*d comlst.17,18
*d comlst.3,9
```

```
        common/scratch/beta(maxsze),d(maxsze)
        common/turb/ dke(maxsze),et(maxsze),pk(maxsze),tke(maxsze)
        common/auxy/ ar(maximp),ax(maximp),az(maximp),cf(maxsze),
       1             cv(maxsze),fdc(maxarr),htc(maxarr),mt(maxsze),
       2             ndc(maxsze),nhc(maxsze),nt(maxsze),qqs(maxarr),
       3             vf(maximp)
        common/flow/ e(maxsze),emu(maxsze),h(maxsze),p(maxsze),qs(maxarr),
       1             rho(maxsze),u(maxsze),v(maxsze),w(maxsze)
        common/heat/ q(maxsze),sp(maxsze),t(maxsze)
        common/stor/ index(maxset),text(maxtxt)
*d comlst.63,66
        common/rem1/ airdg,convj,dragk,efa,emax,eumax,grt,gset,gxt,gzt,
       1             offset,rwind,teql,tess,vsurf,wangl,wangr,wexp,wind,
       2             xwind
        common/movy/ idir,iplno,tmovie,tntrvl
*id acbfx2
*d ncom.3,35
        common/grid/ iq,iqm,iqp,isc,ix,jks,jksm,jsc,kq,kql,kqm,kqp,ksc,
       1             kx,kx1,leaf,ljk,lnr,lwn,mqs,ni,nii,nj,njj,njr,nk,nkk,
       2             nkz,nped,npst,nr1,nrz,page
        common/misc/ iedge,iend,ifield,image,imnmax,in,iout,iplot,iplstp,
       1             iplstr,ipost,iql,iskip,islip,ist,iters,itimes,itmax,
       2             itpd,itpdst,itpdsv,jiio,kard,kase,kiio,kiprbl,kiters,
       3             kitmax,kitmin,kmon,kskip,ksurf,kt20,l30,large,larm,
       4             lasth,lath,lden,lentmx,lfix,liters,litmax,loco,lout,
       5             lp,lr,lrzx,lsa,lsend,lsetmx,lslpmx,lsteps,lt,ltabp,
       6             ltabq,ltc,lx,lz,matref,maxa,maxd,maxmat,maxpv,maxslp,
       7             maxtc,mout,mskip,natab,nbeg,nbtab,ncells,ncon,ndmax,
       8             ndt,next,nfct,nfluid,nfout,nhdmax,nhmax,njs,nmat,
       9             nmaxtc,nocon,nout,nox(lsize),nptabs,nr,ns1,ns2,nsb,
       a            nseal,
       a             nsend,nsetmx,nsinc,nslip,nsolid,nsout,nsp,nspm,nsrc,
       b             nstep,nsteps,ntaba,ntabbt,ntabc,ntabf,ntabft,ntabh,
       c             ntabl,ntabp,ntabq,ntabr,ntabs,ntabv,ntabx,ntaybl,
       d             nvis,nvpr,nvpx,nvpz,nwtab,nx,nz,out
        common/scra/ j01,j02,j03,j04,j05,j06,j07,j08,j09,j10,j11,j12,j13,
       1             j14,j15,j16,j17,j18,j19,j20,j21,j22,j23,j24,j25,j26,
       2             jm18,jp17
        common/ind1/ iqjp,iqkp,jhi,jmim,jmip,jmkm,jmkp,jpim,jpkm,kmim,
       1             kmip,kpim,kqip,kqjp,lpack,lpack2,lpack3,lqpack,mpack,
       2             mpack2,mpack3,npack,npack2,npack3,izero,mwl
        common/ext1/ dmax,dtimax,ihrbl,ipart,iprbl,istack,itab,itabc,
       1             itabd,itabe,itabp1,itabr,ncp,ndp,nef,nep,nprt,nrp,
       2             nsprt,nstart,ntp,tbtprt,tbttd,tdtim,tptim
        common/ext2/ bakup,comp,couple,crashed,dayss,dconv,ebalnc,hconv,
       1             hours,marine,mconv,minut,nhgtab,reduce,skiper,skipit,
       2             skipon,sprop,tconv,tdamp,tensor,xconv,years
        common/ext3/ timunt(2)
*af ,vtype
*comdeck crayfunc
c**************************************************************************-
****
```

```
c**
c**   statement functions here
c**
c**
c**   secondary bit functions
c**
      bshftr(ia1,ia2) = shiftr(ia1,+ia2)
      bshftl(ia1,ia2) = shiftl(ia1,+ia2)
      bmask(ia1) = bshftr(.not.izero,mwl-ia1)
      bclear(ia1,ia2,ia3) =
     $          (ia1.and.(.not.(bshftl((bmask(ia3)),(ia2-1)))))
c     bshftc(ia1,ia2,ia3) = ishftc(ia1,ia2,ia3)
c**
c**   primary bit functions
c**
      bxtrac(ia1,ia2,ia3) =
     $          bshftr((bshftl(bmask(ia3),(ia2-1)).and.ia1),(ia2-1))
      bset(ia1,ia2,ia3,ia4) =
     $   (bclear(ia1,ia2,ia3).or.bshftl((bclear(ia4,ia2+ia3,mwl-ia3)),
     $   (ia2-1)))
c**
c**   bxtrac = extract ia3 bits from ia1 starting at bit ia2
c**            (+ a right shift)
c**   bclear = clear (zero) ia3 bits of ia1 starting at bit ia2
c**   bset   = set ia3 bits of ia1 starting at bit ia2 to the bit
c**         pattern of ia4
c**
c**
c**   grid functions here
c**
c***************************************************************************
****
*id acbfx4
*i tempest.288
      data izero,mwl /0,64/
*call crayfunc
*i setin.14
*call crayfunc
*i input.21
*call crayfunc
*i sedinp.17
*call crayfunc
*i ready.7
*call crayfunc
*i celtyp.11
*call crayfunc
*i tabset.20
*call crayfunc
*i indexr.14
*call crayfunc
*i indbug.9
*call crayfunc
```

```
*i adindx.7
*call crayfunc
*i vsindx.7
*call crayfunc
*b balset.8
*call crayfunc
*i getset.7
*call crayfunc
*b cgrid.7
*call crayfunc
*i twix.8
*call crayfunc
*i pinit.7
*call crayfunc
*i tymset.10
*call crayfunc
*b mosrc.8
*call crayfunc
*b excute.17
*call crayfunc
*b moment.7
*call crayfunc
*b momxt.7
*call crayfunc
*b strain.7
*call crayfunc
*b patter.9
*call crayfunc
*b solvel.15
*call crayfunc
*b balanc.13
*call crayfunc
*b solvet.14
*call crayfunc
*b solvef.15
*call crayfunc
*i height.9
*call crayfunc
*b condif.10
*call crayfunc
*b kemod.8
*call crayfunc
*b pkay.8
*call crayfunc
*b spectr.13
*call crayfunc
*b semble.7
*call crayfunc
*b cemble.10
*call crayfunc
*b visset.7
*call crayfunc
```

```
*b bouset.8
*call crayfunc
*b denary.7
*call crayfunc
*b viscos.7
*call crayfunc
*b propst.9
*call crayfunc
*b librry.10
*call crayfunc
*i output.78
*call crayfunc
*b arout.11
*call crayfunc
*b mapper.8
*call crayfunc
*b lcmmap.8
*call crayfunc
*i monitr.10
*call crayfunc
*b monout.10
*call crayfunc
*b massum.10
*call crayfunc
*b masvol.10
*call crayfunc
*b locati.5
*call crayfunc
*i mvydmp.34
*call crayfunc
*b solvex.15
*call crayfunc
*id acbfx5
*af ,vtype
*comdeck intgvar
      integer sdbug,grdindx,isub
      integer bshftr,bshftl,bmask,bclear,bxtrac,bset
*comdeck realvar
      real hdbug(100)
*i tempest.53
*call intgvar
*call realvar
*i setin.6
*call intgvar
*call realvar
*i input.7
*call intgvar
*call realvar
*i sedinp.2
*call intgvar
*call realvar
*i ready.2
```

```
*call intgvar
*call realvar
*i celtyp.6
*call intgvar
*call realvar
*i tabset.11
*call intgvar
*call realvar
*i indexr.6
*call intgvar
*call realvar
*i indbug.4
*call intgvar
*call realvar
*i adindx.2
*call intgvar
*call realvar
*i vsindx.2
*call intgvar
*call realvar
*i balset.2
*call intgvar
*call realvar
*i getset.2
*call intgvar
*call realvar
*i cgrid.2
*call intgvar
*call realvar
*i twix.2
*call intgvar
*call realvar
*i plabak.2
*call intgvar
*call realvar
*i pinit.2
*call intgvar
*call realvar
*i tymset.2
*call intgvar
*call realvar
*i mosrc.3
*call intgvar
*call realvar
*i excute.2
*call intgvar
*call realvar
*i moment.2
*call intgvar
*call realvar
*i momxt.2
*call intgvar
```

```
*call realvar
*i strain.2
*call intgvar
*call realvar
*i patter.4
*call intgvar
*call realvar
*i solvel.2
*call intgvar
*call realvar
*i balanc.2
*call intgvar
*call realvar
*i solvet.2
*call intgvar
*call realvar
*i solvef.2
*call intgvar
*call realvar
*i efield.2
*call intgvar
*call realvar
*i height.2
*call intgvar
*call realvar
*i condif.2
*call intgvar
*call realvar
*i kemod.2
*call intgvar
*call realvar
*i pkay.3
*call intgvar
*call realvar
*i spectr.5
*call intgvar
*call realvar
*i semble.2
*call intgvar
*call realvar
*i rebal.4
*call intgvar
*call realvar
*i efbal.6
*call intgvar
*call realvar
*i syclic.2
*call intgvar
*call realvar
*i cemble.5
*call intgvar
*call realvar
```

```
*i efconn.5
*call intgvar
*call realvar
*i visset.2
*call intgvar
*call realvar
*i xand.2
*call intgvar
*call realvar
*i xand2.2
*call intgvar
*call realvar
*i xoffal.2
*call intgvar
*call realvar
*i xoby.2
*call intgvar
*call realvar
*i xens.2
*call intgvar
*call realvar
*i xhore.2
*call intgvar
*call realvar
*i xurf.2
*call intgvar
*call realvar
*i xilt.2
*call intgvar
*call realvar
*i xlay.2
*call intgvar
*call realvar
*i xilcla.2
*call intgvar
*call realvar
*i xedhis.2
*call intgvar
*call realvar
*i bouset.2
*call intgvar
*call realvar
*i denary.2
*call intgvar
*call realvar
*i viscos.2
*call intgvar
*call realvar
*i propst.2
*call intgvar
*call realvar
*i librry.2
```

```
*call intgvar
*call realvar
*i output.2
*call intgvar
*call realvar
*i sedout.2
*call intgvar
*call realvar
*i arout.3
*call intgvar
*call realvar
*i mapper.3
*call intgvar
*call realvar
*i lcmmap.3
*call intgvar
*call realvar
*i monitr.2
*call intgvar
*call realvar
*i monout.5
*call intgvar
*call realvar
*i massum.5
*call intgvar
*call realvar
*i masvol.5
*call intgvar
*call realvar
*i locati.3
*call intgvar
*call realvar
*i ierror.2
*call intgvar
*call realvar
*i error.2
*call intgvar
*call realvar
*i banner.2
*call intgvar
*call realvar
*i dattim.25
*call intgvar
*call realvar
*i second.9
*call intgvar
*call realvar
*i mvydmp.24
*call intgvar
*call realvar
*i solvex.7
*call intgvar
```

```
*call realvar
*id subdbug
*d dbswtch0.1
      common/ex8/ idbug(200),sdbug(100),isub(100)
*id nfix29s
*/
*/      update changes to fix version 29
*/      tempest, cray version n, mod 29.3 on 31mar87.
*/      used to create version n, mod30.0 on 21may87
*/
*/ *************************************************** verson ***********
*/
*/      update current version and mod number
*i tempest.45
c**          creation run    - mod 30.0 -     may  19,   1987         **
*d verson.2,3
      data machin,verson,(datver(j),j=1,2),modnum,(datmod(i),i=1,2)
     1           /"cray","n   ","jul ","1985","30.0","may ","1987"/
*/ ***************************************************** common *********
*/
*/      generic common block changes
*/
*/ **********************************************************************
*/
*/   problem number: n29.#
*/
*/     problem:
*/      finder:  Loren Eyler (pnl)  00mon86
*/       fixer:  Loren Eyler
*/
*/ **********************************************************************
*/
*/   problem number: n29.1
*/
*/     problem:  tabulated props not working correctly due to if statement
*/      finder:  Loren Eyler (pnl)  13jul86
*/       fixer:  Loren Eyler
*/
*d tabset.259
      if((k/itab)*itabp1.ge.ktab(1)) go to 480
*d input.731
      if(prop(7,ii).eq.0..and.kb.eq.const) go to 200
*/ **********************************************************************
*/
*/   problem number: n29.2
*/
*/     problem:  incorrect temp dependence in brine properties
*/      finder:  Don Trent (pnl)  14nov86
*/       fixer:  Loren Eyler
*/
*d librry.102
      sigmat   = (sig0+.1324)*(1.-a+b*(sig0-.1324))-sumt
```

```
*d denary.46
      sigmat   = (sig0+.1324)*(1.-a+b*(sig0-.1324))-sumt
*/
*/   miscellaneous cleanup
*/
*d tempest.732
      units(2,31) = "mtrs"
*d indbug.486
     d,6x,"table" //)
*d balset.3
      integer length(lax),cicle(3)
*d plabak.8
      integer    unitq1,unitq2
*d plabak.5,6
      integer    vdx(3),cdx(3),wprt(3)
      integer    typr(3),typz(3),typx(3)
*d monitr.91
      write(out,1024) mint(1),ksurf,mint(3),
     *                mint(4),ksurf,mint(6),
     *                mint(7),ksurf,mint(9),
     *                mint(10),ksurf,mint(12)
*d monitr.270,271
     1 ,t30,"specie  c(",i2,") - - ",2a4,":  mass fraction - - - cmf",/
     2 ,t59,"volume fraction - - cvf"/)
*/
*/     incorrect argument list
*d solvef.74
      call cycle(1,m,lsize,am,ac,ap,bn,tend)
*i ready.204
      if(.not.contrl(61)) goto 180
*i ready.218
  180 continue
*/ **********************************************************************
*/
*/   problem number: n29.3
*/
*/     problem:  cyclic solver in subroutine solvet incorrect
*/      finder:  Don Trent (pnl)  27mar87
*/       fixer:  Don Trent
*/
*i solvet.54
      amm      = am(m)
      acm      = ac(m)
*i solvet.58
      ac(k)    = am(k)*ac(k-1)*bet(k)
*d solvet.68,69
```

```
*/    name = c_tempst_nb1.uz2  (best front-tracker logic 8/1/87 )
*/
*/
*id mjbfix1
*/
*/
*/
*/
*/     name = c_tempst_nb1.ux2
*/
*/     name = c_tempst_nb1.upx
*/
*/          stored on path = /800320/tempest/n29/ctss/b1
*/
*/     these updates are debug updates for the n29b1 version of tempest
*/
*d tempest.326,363
c**   print teapot banner
      call banner
      acard = "xxxx"
      kard = 0
  100 continue
      read(in,10000) inrec
      kard     = kard+1
      write(out,10001) inrec,kard
      acard = inrec(1:4)
      aopt = inrec(6:9)
      zopt = inrec(77:80)
      if(acard.eq."****".or.zopt.eq."****") go to 100
      if(acard.eq."grid") go to 105
      if(acard.eq."size") go to 105
      if(acard.eq."time") go to 110
      if(acard.eq."prnt") go to 115
      if(acard.eq."rest") go to 120
      if(acard.eq."pres") go to 125
      if(acard.eq."misc") go to 130
      if(acard.eq."post") go to 135
      if(acard.eq."seal") go to 140
      if(acard.eq."temp") go to 145
      if(acard.eq."elec") go to 150
      if(acard.eq."movy") go to 160
      if(acard.eq."dbug") go to 170
      if(acard.eq."frnt") go to 190
      if(acard.eq."    ".and.xcheck(1)) go to 200
      if(acard.eq."----") go to 200
      go to 100
*i tempest.367
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.376
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.390
      read(inrec,10002) acard,(data(i),i=1,14),idat
```

```
*i tempest.406
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.415
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.429
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.446
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.456
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.468
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.477
      read(inrec,10002) acard,(data(i),i=1,14),idat
*i tempest.490
      read(inrec,10002) acard,(data(i),i=1,14),idat
*b tempest.498
  190 continue
c**
c**   front-tracker card
c**
      read(inrec,10002) acard,(data(i),i=1,14),idat
      kfrnt = int(data(1))
      ncpf  = int(data(2))
      dzu   = int(data(3))
      dzl   = int(data(4))
      nz = nz + 2*ncpf + 1
      if(kfrnt.le.0) kfrnt = 99999
      go to 100
*/
*/     version subroutine
*/
*af ,tempest
*deck version
      subroutine version
c
c  this routine was written primarily  to create a banner page for the
c  tempst code.  it will be used to provide information on the version
c  number of tempst currently in use, update information, and applicabl
c  documentation.  author: cl bartley 4/15/86
c
      character *1 star,minus
c
      star = "*"
      minus = "-"
c
      idat = 6
c
c
c  change all writes to unit 9 to idat later
c
      write(idat,50)
```

```
50     format(/)
c
       do 150 j=1,5
       write(idat,100) (star,i=1,122)
100    format(3(5x 122a1/))
150    continue
c
       do 250 j=1,2
       write(idat,200)(star,i=1,40)
200    format(5x,20a1,82x,20a1/)
250    continue
c
       write(idat,300) (star,i=1,40)
300    format(5x,20a1,34x,"t e m p e s t",35x,20a1)
c
       write(idat,400) (star,i=1,40)
400    format(//5x,20a1,23x,"code version and update information",24x,
      120a1)
c
       write(idat,500) (star,i=1,40)
500    format(/5x,20a1,22x,"written for lasl ctss by m. j. budden",
      123x,20a1)
c
       write(idat,600) (star,i=1,40)
600    format(/5x,20a1,11x,"converted to vax 11/780 by (  not  yet    ",
      1"   not   yet    )",12x,20a1)
c
       write(idat,700) (star,i=1,40)
700    format(/5x,20a1,26x,"applicable documents to follow",26x,20a1)
c
       write(idat,800) (star,i=1,40)
800    format(/5x,20a1,22x,"ctss mass version number ( b1  )",28x,20a1)
c
       write(idat,810) (star,i=1,40)
810    format(/5x,20a1,5x,"dir = /800320/tempest/n29/b1",49x,20a1)
       write(idat,820) (star,i=1,40)
820    format(/5x,20a1,10x,"tempst_nb1.src   historian source file",34x,
      $20a1)
       write(idat,830) (star,i=1,40)
830    format(/5x,20a1,10x,"tempst_nb1.opl   historian old program librar
      $y",26x,20a1)
       write(idat,840) (star,i=1,40)
840    format(/5x,20a1,10x,"tempst_nb1.lib   ctss/cft  object library",
      $31x,20a1)
       write(idat,850) (star,i=1,40)
850    format(/5x,20a1,10x,"tempst_nb1.for   historian fortran compile
      $file",23x,20a1)
       write(idat,860) (star,i=1,40)
860    format(/5x,20a1,10x,"tempst_nb1.lis     cft fortran list file",32x,
      $20a1)
       write(idat,810) (star,i=1,40)
       do 950 j=1,5
      write(idat,200) (star,i=1,40)
950   continue
c
      do 1050 j=1,5
      write(idat,100) (star,i=1,122)
1050  continue
      write(idat,1200)
1200  format(1h1)
      write(idat,50)
c
      do 1250 j=1,5
      write(idat,100) (star,i=1,122)
1250  continue
c
      do 1300 j=1,2
      write(idat,200)(star,i=1,40)
1300  continue
c
      write(idat,1310) (star,i=1,20)
      write(idat,1311) (star,i=1,20)
1310  format(5x,20a1,5x,"preceeding version a1")
c     write(idat,1312) (star,i=1,20)
1311  format(5x,20a1,10x,"this version is for glass pouring. it has comm
     $ents and multiple title cards on the first cards.")
1312  format(5x,20a1,10x,"these cards are identified by the pattern
     $"****" in columns 1 - 4 of each data card.")
      write(idat,1200)
      return
      end
*i tempest.291
      call version
*/   name = upq
*/
*/     beginning of tempest multi grid updates
*/
*/     compatible with:
*/          c_tempst_nb1.xxx
*/
*i tempest.371
      igrid = int(data(13))
      orign = data(12)
      if (igrid.eq.0) igrid = 1
      ngrid = max(ngrid,igrid)
      grdtyp = int(data(11))
      ii = (igrid - 1)*ispgrd + 1
      grdindx(ii)   = igrid
      grdindx(ii+1) = grdtyp
      grdindx(ii+2) = orign
      grdindx(ii+3) = 0
      grdindx(ii+4) = 0
      grdindx(ii+5) = nr
      grdindx(ii+6) = nz
      grdindx(ii+7) = nx
      grdindx(ii+8) = nr + nz + nx
      grdindx(ii+9) = nr*nz*nx
*i tempest.981
c**
c**   finish loading grdindx array
c**
      do 810 igrid = 2,ngrid
      ii = (igrid-1)*ispgrd + 1
      grdindx(ii+3) = grdindx(ii+9) + grdindx(ii-1)
      grdindx(ii+4) = grdindx(ii+8) + grdindx(ii-2)
  810 continue
*i plabak.52
      write(out,90004)
      write(out,90005)
      do 10 igrid = 1,ngrid
      ii = (igrid-1)*ispgrd + 1
      ll = ii-1
      write(out,90006) (grdindx(ll+m),m=1,10)
   10 continue
      write(out,90007)
*i plabak.758
90004 format(///,2x,"print out grdindx array",///)
90005 format(//,2x,"grid #",2x,"grid type",2x,"origen",2x,"offset #1",
     1       2x,"offset #2",2x,"nr",2x,"nz",2x,"nx",2x,"nr+nz+nx",2x,
     2       "nr*nz*nx",/)
90006 format(2x,f5.0,2x,f5.0,2x,1pe12.5,2x,f5.0,2x,f5.0,2x,f5.0,
     1       2x,f5.0,2x,f5.0,2x,f5.0,2x,f5.0)
90007 format(///)
*/     name = upz2
*/
*/     beginning f-tracker/multi-grid updates
*/
*/
*/     compatible with c_tempst_nb1.xxx
*/
*id mltgrd1
*d tempest.842,926
*d tempest.942,944
*af ,tempest
*deck gridi
      subroutine gridi(mode,iflag)
*call intgvar
*call realvar
*call facts
*call comlst
*call ncom
*call vtype
*call sedt
*call crayfunc
c**
c**
```

```
c**
      if(mode.gt.9.or.mode.lt.1) go to 9999
      go to (1000,2000,3000,4000,5000,6000,7000,8000,9000),mode
c**
c**   set array size and other index parameters
c**
 1000 continue
      nj       = nr - 1
      nk       = nz - 1
      ni       = nx - 1
      if(nj.eq.0) nj = 1
      if(nk.eq.0) nk = 1
      if(ni.eq.0) ni = 1
      if(ni.le.1) twodim = .true.
      njr      = nj
      nkz      = nk
      nix      = ni
      nrz      = nr
      jsc      = lr*nr
      ksc      = jsc + lz*nz
      isc      = ksc + lx*nx
      jks      = nx*nr*nz
      kq       = jks
      iq       = jks*2
      ljk      = nr*nz
      lwn      = 1
      lnr      = nr
      if(nr.eq.1) lwn = 0
      if(nz.eq.1) lnr = 0
      if(nx.eq.1) ljk = 0
      kx       = nr + nz
      ix       = kx + nx
      kqm      = kq - lnr
      kqp      = kq + lnr
      iqm      = iq - ljk
      iqp      = iq + ljk
      jm18     = j18 - lr*lwn
      jp17     = j17 + lr*lwn
      kqjp     = kq + lwn
      kqip     = kq + ljk
      iqjp     = iq + lwn
      iqkp     = iq + lnr
      jmip     = ljk - lwn
      jmkp     = lnr - lwn
      jpkm     = lwn - lnr
      kmip     = ljk - lnr
      jpim     = lwn - ljk
      kpim     = lnr - ljk
      jmkm     =-lwn - lnr
      jmim     =-lwn - ljk
      kmim     =-lnr - ljk
      mqs      = jks*3
```

```
     npst     = ljk + lnr + lwn
     kql      = kq + npst
     iql      = iq + npst
     nped     = jks - npst
     jksm     = jks - 1
     page     = (63/(nz + 5))
     leaf     = (63/(nz + 2))
     nr1      = nr + 1
     kx1      = kx + 1
     knox     = max(nr,nx,nz)
     lrzx     = knox
     njj      = 2
     nkk      = 2
     nii      = 2
     if(nr.gt.1) go to 620
     nrz      = nz
     njr      = 1
     njj      = 1
     tdfr     = 0.
     tdr      = 0.
     map(1,1) = 0
     map(1,2) = 0
 620 if(nz.gt.1) go to 625
     nkz      = 1
     nkk      = 1
     tdz      = 0.
     tdfz     = 0.
     map(2,3) = 0
     map(2,4) = 0
 625 if(nx.gt.1) go to 630
     nix      = 1
     nii = 1
     tdx = 0.0
     tdfx     = 0.
     map(3,5) = 0
     map(3,6) = 0
 630 continue
     njs      = njj + int(cyl)
     if(leaf.eq.0) leaf = 1
     if(page.eq.0) page = 1
     go to 9999
2000 continue
3000 continue
4000 continue
5000 continue
6000 continue
7000 continue
8000 continue
9000 continue
9999 continue
     return
     end
```

```
*b tempest.982
      write(out,3001)
      write(out,3002)
      do 815 igrid = 1,ngrid
      ii = (igrid - 1)*ispgrd + 1
      write(out,3003) igrid,(grdindx(ii+l),l = 0,9)
  815 continue
*b tempest.984
      call gridi(1,1)
*i tempest.1047
3001  format(//,2x,"grid index array",//)
3002  format(2x,"grid #",2x,"grid type",2x,"origin",2x,
     $      "offset",2x,"offset",2x,"nr",2x,"nz",2x,"nx",2x,
     $      "ndelta",2x,"ncells")
3003  format(2x,10(i5,2x),2x)
10000 format(a80)
10001 format(a80,10x,i5)
10004 format(a4,1x,14f5.0,i5)
10003 format(a4,1x,a4,1x,6(2x,a8))
*/
*/     improved input
*/
*i tempest.351
      if(xcard.eq."grid") go to 105
*i input.3
      character inrec*80,blank5*5
      character*5 char5(16)
      character*4 acard,aopt,bopt,zopt
*d input.212,214
  100 continue
      read(in,10000) inrec
      kard = kard+1
      if(debug(22)) write(out,10002) inrec,kard
      acard = inrec(1:4)
      aopt = inrec(6:9)
      bopt = inrec(11:15)
      zopt = inrec(77:80)
      if(acard.eq."****".or.zopt.eq."****") go to 100
      if(acard.eq."grid") go to 105
      backspace in
      go to 109
  105 continue
      read(inrec,10001) ikard,igkind,igrid
      if(igrid.le.ngrid) go to 106
      write(out,90000) igrid,ngrid
      bombed = .true.
      igrid = ngrid
  106 continue
  109 continue
      read(in,10003) ndata,jm,km,im,kb,kp,lc,kind,sys
*d input.470
  200 continue
```

```
      read(in,10000) inrec
      kard     = kard+1
      write(out,10002) inrec,kard
      acard = inrec(1:4)
      aopt = inrec(6:9)
      bopt = inrec(11:15)
      zopt = inrec(77:80)
      if(acard.eq."****".or.zopt.eq."****") go to 200
      if(acard.eq."grid") go to 205
      backspace in
      go to 209
  205 continue
      read(inrec,10001) ikard,igkind,igrid
      if(igrid.le.ngrid) go to 206
      write(out,90000) igrid,ngrid
      bombed = .true.
      igrid = ngrid
  206 continue
  209 continue
      read(in,10004) data,ji,kii,ki,iii,ii,kb,kp,lc,type,sys
*d input.472
*d input.485
*i input.1440
10000 format(a80)
10001 format(2(a4,1x),14f5.0)
10002 format(10x,a80,10x,i5)
10003 format(10i5,3i5,2a1,i1,i2,a2)
10004 format(10f5.0,i5,i2,i3,i3,i2,2a1,i1,i2,a2)
90000 format(///,2x,"error igrid > ngrid",2x,"igrid=",i5,2x,"ngrid=",
     1       i5,///)
90001 format(2x,"line=",i5,2x,"nn =",i5,2x,"l=",i5,2x,"m=",i5,2x,
     $"n=",i5,2x,"kfrnt=",i5,2x,"ncpf=",i5,2x,"mrezon=",i5,2x,
     $"iadd=",i5,2x,"mm=",i5)
90002 format(2x,"line=",i5,2x,"nn=",i5,2x,"l =",i5,2x,"m=",i5,2x,
     $"n=",i5,2x,"kfrnt=",i5,2x,"ncpf=",i5,2x,"mrezon=",i5,2x,
     $"iadd=",i5,2x,"nhc(nn)=",i5)
90003 format(2x,"line= ",i5,2x,"nn=",i5,2x,"l=",i5,2x,"m=",i5,2x,
     $"n=",i5,2x,"kfrnt=",i5,2x,"ncpf=",i5,2x,"mrezon=",i5,2x,
     $"iadd=",i5,2x,"ndc(nn)=",i5)
*/
*/     front-tracker changes to input
*/
*id ftrack1
*i tempest.300
      kfrnt = 99999
      ncpf = 0
*i input.242
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 111
      m2 = mrezon + 2*ncpf + 1
  111 continue
      do 114 mm = m1,m2
      nn        = locate(l,mm,n)
*d input.243
      line = 243
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,mm
*i input.236
      iadd = im/100
      im = im - 100*iadd
*d input.260
      if(nt(nn).ne.25) go to 114
*i input.272
  114 continue
*i input.328
      iadd = im/100
      im = im - 100*iadd
*i input.331
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 131
      m2 = mrezon + 2*ncpf + 1
  131 continue
      do 134 mm = m1,m2
      nn        = locate(l,mm,n)
*d input.332
*i input.333
      line = 333
      if(idbug(002).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90002) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,
     $       nhc(nn)
  134 continue
*i input.382
      iadd = im/100
      im = im - 100*iadd
*i input.385
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 141
      m2 = mrezon + 2*ncpf + 1
  141 continue
      do 144 mm = m1,m2
      nn       = locate(1,mm,n)
*d input.386
*i input.387
      line = 387
      if(idbug(003).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90003) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,
     $      ndc(nn)
*i input.389
  144 continue
*i input.413
      if(ndata(2).gt.kfrnt) ndata(2) = ndata(2) + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(ndata(2).eq.kfrnt) ndata(2) = ndata(2) + iadd
      if(ndata(2).lt.kfrnt) ndata(2) = ndata(2)
      if(ndata(2).gt.kfrnt) ndata(2) = ndata(2) + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(ndata(5).eq.kfrnt) ndata(5) = ndata(5) + iadd
      if(ndata(5).lt.kfrnt) ndata(5) = ndata(5)
      if(ndata(5).gt.kfrnt) ndata(5) = ndata(5) + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(ndata(8).eq.kfrnt) ndata(8) = ndata(8) + iadd
      if(ndata(8).lt.kfrnt) ndata(8) = ndata(8)
      if(ndata(8).gt.kfrnt) ndata(8) = ndata(8) + 1 + 2*ncpf
*d input.570
      if(kb.eq."*") go to 246
*d input.575
      line = 575
      l = n
      if(kin.gt.kfrnt) l = n + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = ii
      if(kin.eq.kfrnt) l = n + iadd
      if(kin.lt.kfrnt) l = n
      dx(l) = data(kin)*dkind(1,nit)*dkant
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90100) line,l,n,kin,kfrnt
90100 format(2x,"line= ",i5,2x,"l=",i5,2x,"n=",i5,2x,"kin=",
     1 i5,2x,"kfrnt=",i5)
*i input.748
c**                                                         (lc    = 0)
*i input.750
      if(lc.eq.1) go to 276
      if(lc.eq.2) go to 278
*i input.769
      line = 769
      if(idbug(003).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd
*i input.793
c**                                                         (lc    = 0)
```

```
*i input.795
      if(lc.eq.1) go to 296
      if(lc.eq.2) go to 298
*i input.778
c**   set up u,v,w and t boundary velocity component input  (type = 7)
c**                                                         (lc   = 1)
  276 continue
      if(contrl(6).and.kp.ne.mrl) go to 200
      j1        = data(5)
      j2        = data(6)
      k1        = data(7)
      k2        = data(8)
      i1        = data(9)
      i2        = data(10)
      if(j1.eq.0) j1 = 1
      if(j2.eq.0) j2 = 1
      if(k1.eq.0) k1 = 1
      if(k2.eq.0) k2 = 1
      if(i1.eq.0) i1 = 1
      if(i2.eq.0) i2 = 1
      if(j1.gt.j2.or.j2.gt.nr) call error(kard,4,bombed)
      if(k1.gt.k2.or.k2.gt.nz) call error(kard,5,bombed)
      if(i1.gt.i2.or.i2.gt.nx) call error(kard,6,bombed)
      do 277 l = j1,j2
      do 277 m = k1,k2
      do 277 n = i1,i2
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 2771
      m2 = mrezon + 2*ncpf + 1
 2771 continue
      do 2772 mm = m1,m2
      nn        = locate(l,mm,n)
      line = 778
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,mm
      u(nn)     = data(1)*dkind(1,nit)
      v(nn)     = data(2)*dkind(1,nit)
      w(nn)     = data(3)*dkind(1,nit)
      if(data(4).ne.0.) t(nn) = tcnvrt(data(4),dkind(2,nit))
 2772 continue
  277 continue
      go to 200
c**   set up tke,dke,et and t boundary vel. component input (type = 7)
c**                                                         (lc   = 2)
  278 continue
      if(contrl(6).and.kp.ne.mrl) go to 200
      j1       = data(5)
      j2       = data(6)
      k1       = data(7)
      k2       = data(8)
      i1       = data(9)
      i2       = data(10)
      if(j1.eq.0) j1 = 1
      if(j2.eq.0) j2 = 1
      if(k1.eq.0) k1 = 1
      if(k2.eq.0) k2 = 1
      if(i1.eq.0) i1 = 1
      if(i2.eq.0) i2 = 1
      if(j1.gt.j2.or.j2.gt.nr) call error(kard,4,bombed)
      if(k1.gt.k2.or.k2.gt.nz) call error(kard,5,bombed)
      if(i1.gt.i2.or.i2.gt.nx) call error(kard,6,bombed)
      do 279 l = j1,j2
      do 279 m = k1,k2
      do 279 n = i1,i2
      m2 = m
      m1 = mrezon
      if(m.ne.kfrnt) go to 2793
      m2 = m + 2*ncpf + 1
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 2793
      m2 = mrezon + 2*ncpf + 1
 2793 continue
      do 2794 mm = m1,m2
      nn       = locate(l,mm,n)
      line = 779
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,mm
      nn       = locate(l,mm,n)
      tke(nn)  = data(1)*dkind(1,nit)*dkind(1,nit)+smallr
      dke(nn)  = data(2)*dkind(1,nit)*dkind(1,nit)+smallr
      et(nn)   = data(3)*dkind(10,nit)+smallr
      if(data(4).ne.0.) t(nn) = tcnvrt(data(4),dkind(2,nit))
 2794 continue
  279 continue
      go to 200
*i input.822
c**   set-up t,q and p initial conditions                       (type = 9)
c**                                                             (lc   = 1)
  296 continue
      if(contrl(6).and.kp.ne.mrl) go to 200
      j1       = data(5)
      j2       = data(6)
      k1       = data(7)
      k2       = data(8)
      i1       = data(9)
      i2       = data(10)
      if(j1.eq.0) j1 = 1
      if(j2.eq.0) j2 = 1
      if(k1.eq.0) k1 = 1
      if(k2.eq.0) k2 = 1
      if(i1.eq.0) i1 = 1
      if(i2.eq.0) i2 = 1
      if(j1.gt.j2.or.j2.gt.nr) call error(kard,8 ,bombed)
      if(k1.gt.k2.or.k2.gt.nz) call error(kard,9 ,bombed)
      if(i1.gt.i2.or.i2.gt.nx) call error(kard,10,bombed)
      do 297 l = j1,j2
      do 297 m = k1,k2
      do 297 n = i1,i2
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 2971
      m2 = mrezon + 2*ncpf + 1
 2971 continue
      do 2972 mm = m1,m2
      nn       = locate(l,mm,n)
      nn       = locate(l,mm,n)
      line = 822
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,mm
      if(data(1).ne.0.)t(nn)  = tcnvrt(data(1),dkind(2,nit))
      if(data(2).ne.0.)q(nn)  = data(2)*dkind(8,nit)/(dkant**3)
      if(data(3).ne.0.)p(nn)  = data(3)*dkind(9,nit)
 2972 continue
  297 continue
      go to 200
c**   set-up tke,dke and et initial conditions              (type = 9)
c**                                                         (lc   = 2)
  298 continue
      if(contrl(6).and.kp.ne.mrl) go to 200
      j1       = data(5)
      j2       = data(6)
      k1       = data(7)
      k2       = data(8)
      i1       = data(9)
      i2       = data(10)
      if(j1.eq.0) j1 = 1
      if(j2.eq.0) j2 = 1
      if(k1.eq.0) k1 = 1
      if(k2.eq.0) k2 = 1
      if(i1.eq.0) i1 = 1
      if(i2.eq.0) i2 = 1
      if(j1.gt.j2.or.j2.gt.nr) call error(kard,8 ,bombed)
      if(k1.gt.k2.or.k2.gt.nz) call error(kard,9 ,bombed)
      if(i1.gt.i2.or.i2.gt.nx) call error(kard,10,bombed)
      do 299 l = j1,j2
      do 299 m = k1,k2
      do 299 n = i1,i2
      mrezon = m
      if(m.gt.kfrnt) mrezon = m + 1 + 2*ncpf
      if(kb.eq."f".and.kp.eq."*") iadd = im
      if(m.eq.kfrnt) mrezon = m + iadd
      if(m.lt.kfrnt) mrezon = m
      m2 = mrezon
      m1 = mrezon
      if(m.ne.kfrnt) go to 2991
      m2 = mrezon + 2*ncpf + 1
 2991 continue
      do 2992 mm = m1,m2
      nn       = locate(l,mm,n)
      nn       = locate(l,mm,n)
      line = 823
      if(idbug(001).eq.1.and.isub(isubn).eq.1)
     $ write(outd,90001) line,nn,l,m,n,kfrnt,ncpf,mrezon,iadd,mm
      if(data(1).ne.0.)tke(nn)= data(5)*dkind(1,nit)*dkind(1,nit)+smallr
      if(data(2).ne.0.)dke(nn)= data(6)*dkind(1,nit)*dkind(1,nit)+smallr
      if(data(3).ne.0.)et(nn) = data(7)*dkind(10,nit)+smallr
 2992 continue
  299 continue
      go to 200
*/
*/     general debug
*/
*i tempest.57
      character*8 dsub(100),asub(6)
      character*80 inrec
      character*4 acard,aopt,bopt,zopt
*i tempest.256
      data(dsub(i),i = 1,100)/
     $          " grdindx"," tempest","   setin","   input",
     $          "  sedinp","   ready","  celtyp","  tabset",
     $          "  indexr","  indbug","  adindx","  vsindx",
     $          "  balset","  getset","   cgrid","    twix",
     $          "  plabak","   pinit","  tymset","   mosrc",
     $          "  excute","  moment","   momxt","  strain",
     $          "  patter","  solvel","  balanc","  solvet",
     $          "  solvef","  efield","  height","  condif",
     $          "   kemod","    pkay","  spectr","  semble",
     $          "   rebal","   efbal","   cycle","  cemble",
     $          "  efconn","  visset","    xand","   xand2",
     $          "  xoffal","    xoby","    xens","   xhore",
     $          "    xurf","    xilt","    xlay","  xilcla",
     $          "  xedhis","  bouset","  denary","  viscos",
     $          "  propst","  librry","  output","   arout",
     $          "  mapper","  lcmmap","  monitr","  monout",
     $          "  massum","  masvol","  locati","  ierror",
     $          "  banner","  dattim","  second","  mvydmp",
     $          "  solvex",27*"xxxxxxxx"/
*i dbswtch1.5
      if(aopt.ne."subs")
     $  read(inrec,10002) acard,(data(i),i=1,14),idat
      if(aopt.eq."subs") go to 176
*i dbswtch1.15
  176 continue
      read(inrec,10003) acard,aopt,(asub(i),i=1,6)
      do 177 i = 1,6
      do 177 n = 1,100
      if(asub(i).eq.dsub(n)) isub(n) = 1
      line = 775
c     write(outd,88888) line,i,n,asub(i),dsub(n),isub(n),asub(i),dsub(n)
88888 format(2x,"line=",i5,2x,"i=",i5,2x,"n=",i5,2x,"asub=",o22,2x,
     1        "dsub=",o22,2x,"isub=",i5,/,32x,"asub=",a8,16x,"dsub=",a8)
  177 continue
      go to 100
*b tempest.289
c**
c**
      isubn = 2
c**
c**
*b gridi.14
c**
c**
      isubn = 1
c**
c**
*b version.11
c**
c**
      isubn = 74
c**
c**
*b input.25
c**
c**
      isubn = 4
c**
c**
*b celtyp.13
c**
c**
      isubn = 7
c**
c**
```

```
*b indexr.16
c**
c**
      isubn = 9
c**
c**
*b balset.9
c**
c**
      isubn = 13
c**
c**
*i plabak.51
c**
c**
      isubn = 17
c**
c**
*b pinit.9
c**
c**
      isubn = 18
c**
c**
*b mosrc.8
c**
c**
      isubn = 20
c**
c**
*b moment.7
c**
c**
      isubn = 22
c**
c**
*b patter.9
c**
c**
      isubn = 25
c**
c**
*b solvel.17
c**
c**
      isubn = 26
c**
c**
*b balanc.14
c**
c**
      isubn = 27
```

```
c**
c**
*b bouset.11
c**
c**
      isubn = 54
c**
c**
*b solvex.15
c**
c**
      isubn = 73
c**
c**
*id fixmosrc
*i getset.74
      if(nsrc.gt.0) call mosrc(0)
*d mosrc.24,29
      if(k1.eq.1.or.k1.eq.3) qqs(n)     = small
      if(k1.eq.2.or.k1.eq.3) qqs(n-lwn)= small
      if(k2.eq.1.or.k2.eq.3) qqs(n+kq)  = small
      if(k2.eq.2.or.k2.eq.3) qqs(n+kqm)= small
      if(k3.eq.1.or.k3.eq.3) qqs(n+iq)  = small
      if(k3.eq.2.or.k3.eq.3) qqs(n+iqm)= small
```

# APPENDIX C

## HISTORIAN SOFTWARE UTILITY

# APPENDIX C

## HISTORIAN SOFTWARE UTILITY

This appendix describes the file-naming convention, the location of stored files, and the function of HISTORIAN (which is used as an automated version tracking mechanism). This is provided to enhance the traceability requirements for code development and to facilitate restart of the project.

The important files are stored on the LANL CFS in a simple tree structure. The base version of TEMPEST for the FY87 work is the N29 version, and the starting point is the A0 version of the N29 version. The CFS storage tree is shown in Figure C.1.

The dotted directories are the ones used for this work. Inside each directory are special files indicating the type of file and the version number. These are:

1. C_TEMPST_NA0.SRC = HISTORIAN Source file
2. C_TEMPST_NA0.OPL = HISTORIAN Old Program Library
3. C_TEMPST_NA0.LIB = Relocatable Object Library
4. C_TEMPST_NA0.FOR = FORTRAN file, input to compiler
5. C_TEMPST_NA0.LIS = FORTRAN Listing file with cross reference map
6. C_TEMPST_NA0.EXE = Executable version, absolute binary
7. C_TEMPST_NA0.UPD = A0 UPdate change file for UPDATE or HISTORIAN
8. C_TEMPST_NB0.UPD = B0 Update change file for UPDATE or HISTORIAN

The file-naming convention is very simple and makes use of the 16-character limit for CFS file names. The leading C indicates that these are all CRAY versions of the code. The underscores set off the 6-character name of the code (TEMPEST is abbreviated TEMPST), and also delimit the 3-character version designation NA0. Version N29 is implicitly understood because these files are stored in the A0 subdirectory of the N29 subdirectory as shown in Figure C.1. Finally, the file type is separated from the rest of the name by a period followed by a 3-character designation.

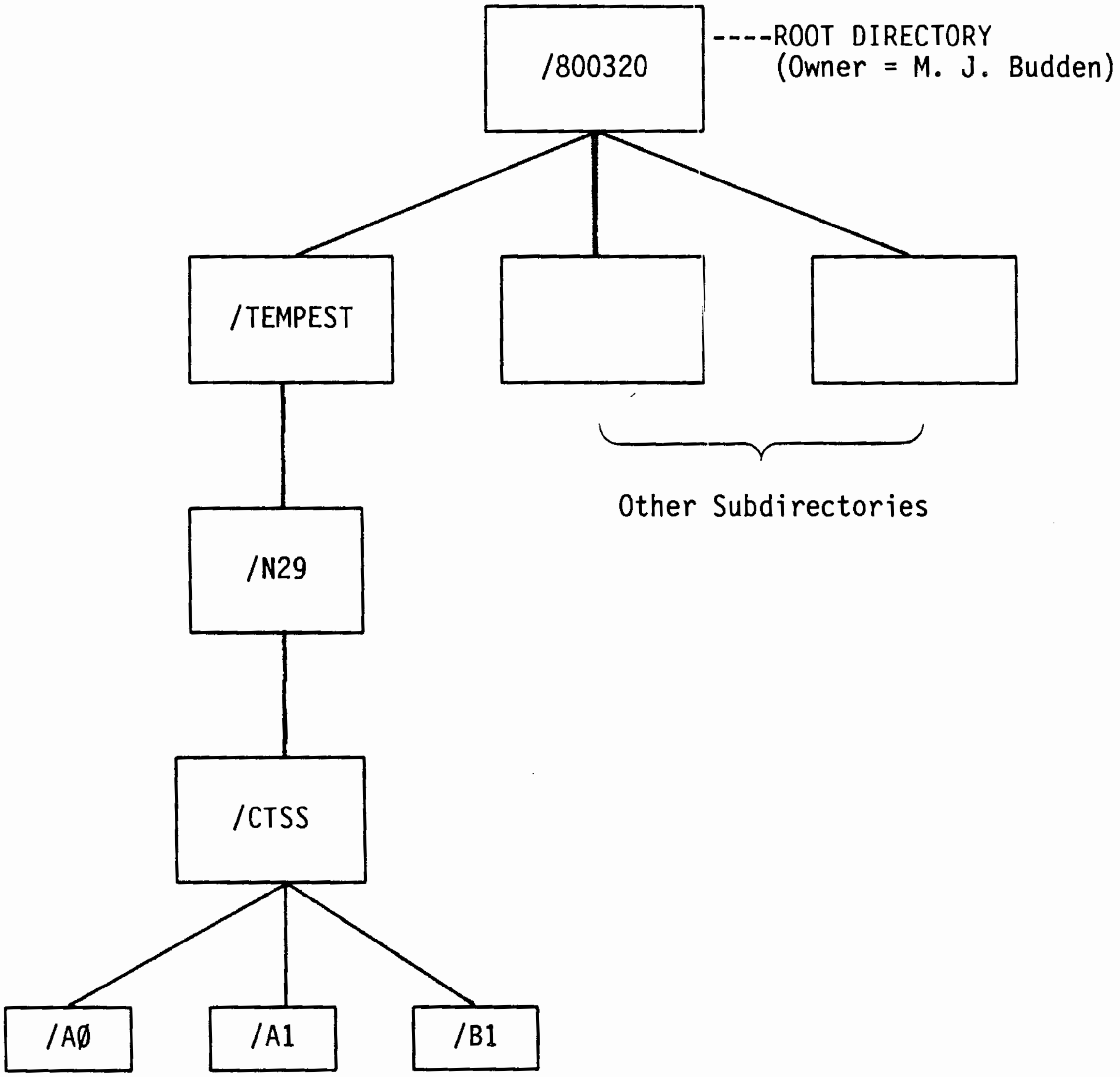


FIGURE C.1. Directory Tree Structure on LANL Common File System Where Appropriate TEMPEST Versions Are Stored

Files 1, 2, 7, and 8 above are special files used by HISTORIAN. HISTORIAN fulfills its traceability function by assigning a unique identifier composed of characters and consecutive integers to each line of code. Directives about inserting new code and deleting existing lines can then be made by referring to the unique identifier of each line. One advantage of this version tracking method is that succeeding versions can be maintained by carrying only the change set, which probably would not exceed a few thousand lines of code. This is a big savings when compared to the 30,000-plus lines of code in each version of TEMPEST. The changes from one version to another are also immediately visible, so the user need not search a 30,000-line FORTRAN source program to understand the differences between versions.

To begin to put a new code under configuration management using HISTORIAN, one must first create a HISTORIAN source file. This is file number 1, with the .SRC extension to indicate the file type. A HISTORIAN source file looks just like an ordinary FORTRAN program except that subroutines and the main program are preceded by *DECK cards to define the HISTORIAN identifiers. Also, COMMON blocks are preceded by *COMDECK directives. The *COMDECK feature accomplishes what the INCLUDE statement does on some FORTRAN-77 extended compilers. Figure C.2 shows parts of the .SRC file for TEMPEST.

When the .SRC file is used as input to HISTORIAN, a complete FORTRAN file can be, and is usually, one of the outputs from HISTORIAN. This file is ready to go to the compiler as input. Figure C.3 shows parts of the .FOR file that comes out of HISTORIAN. Notice that it looks like an ordinary FORTRAN source file, except that columns 80 through 96 contain the HISTORIAN-generated identification for each line. The FORTRAN compiler disregards all information beyond column 72 on each line.

Figure C.4 shows an example .UPD change file. The most often-used directives are:

- *I, which means insert after
- *B, which means insert before
- *D, which means delete and insert after.

Note that the *I in columns 1 and 2 is interpreted by HISTORIAN or UPDATE as a comment card and that the *ID card defines the identifier for all subsequent changes.

HISTORIAN provides the traceability function that is important to satisfy requirements in PNL-MA-70. It also provides a convenient framework for a team of programmers to work with, so that changes made by one individual are more easily understood by another individual and less likely to adversely impact his unique efforts. Finally, HISTORIAN provides the continuity necessary for development on a big program. This makes the project somewhat independent of individual staff members and therefore minimizes problems arising from changes in project staffing.

```
*comdeck facts
      parameter( maxsze = 2500)
      parameter( lsize  = 102)
      parameter( maxsc  = 2600)
      parameter( maxtxt = 4000)
      parameter( maxset =  2000)
c**
      parameter( maxarr = 3*maxsze)
*comdeck comlst
c**
      common/lcm/beta(maxsze),d(maxsze),p(maxsze),q(maxsze),h(maxsze),
     1           et(maxsze),pk(maxsze),rho(maxsze),tke(maxsze),
     2           dke(maxsze),cv(maxsze),cf(maxsze),emu(maxsze),
     3           sp(maxsze)
      common/scm/u(maxsze),v(maxsze),w(maxsze),t(maxsze),e(maxsze),
     1           nt(maxsze)
      common/lca/qs (maxarr),qqs(maxarr),fdc(maxarr),htc(maxarr)
*comdeck verson
      data machin,verson,(datver(j),j=1,2),modnum,(datmod(i),i=1,2)
     1           /"cray","npsl","jul ","1985","29.1","july","1986"/
c**                                                                   **
c**                                                                   **
*deck tempest
      dimension  alist(50),blist(100),clist(30),dlist(30),elist(15),
     1           flist(15),option(15),data(15),ufps(2,35),umks(2,35),
     2           xcheck(11),card(15),alph(3)
      dimension  dat(15)
      logical    xcheck,jiio,kiio,bombed
      integer    tymunt,ufps,umks
*call facts
*call comlst
*call ncom
*call sedt
*call vtype
c**
*call verson
                           •

                           •

                           •


  585 continue
      lskan(11)= lskan(10)+large
c**
c**   set array size and other index parameters
c**
      nj       = nr-1
      nk       = nz-1
      ni       = nx-1
      if(nj.eq.0) nj = 1
```

FIGURE C.2. HISTORIAN Source File Example

```
      if(nk.eq.0) nk = 1
      if(ni.eq.0) ni = 1

                      ●

                      ●

                      ●

                      ●

                      ●

                      ●

                      ●

                      ●
*deck input
      subroutine input
      dimension  data(10),ndata(10),unit(5)
c**
c**   the following equivalence is inserted for computer
c**   compatibility
c**
      dimension qmnt(1)
      equivalence (qmnt,qmont)
      common /ver/machin,verson,datver(2),modnum,datmod(2)
      common/lst/datal(20),lcard
      integer    gas,sol,const,vari,type
      logical    bombed,tonly,lcard,intdat,reldat
*call facts
*call comlst
*call ncom
*call sedt
*call vtype
      data       mrl,mxl,mzl,gas,liq,sol,const,vari,mix,kpc/
     1           lhr,lhx,lhz,lhg,lhl,lhs,lhc   ,lhv ,lhm,lhp/
      data       unit/2h   ,2hes,2hsi,2hin,2hcm/
c**
c**   note - values of mt(n) are packed in this subroutine
c**          unpacking is executed in subroutine indexr

                      ●

                      ●

      ndc(nn)  = ndata(1)*npack2+ndata(2)*npack +ndata(3)
      ndcmax   = max(ndcmax,ndr,ndz,ndx)
      lslpmx   = max(lslpmx,nsr,nsz,nsx)
  145 continue
      go to 100
c**   set up monitor indices                                  (kind = 5)
  150 continue
```

FIGURE C.2. (Contd)

```
    do 152 n = 1,10
    mint(n)  = ndata(n)
    if(mint(n).eq.0) mint(n) = 1
152 continue
```

FIGURE C.2. (Contd)

unique efforts. Finally, HISTORIAN provides the continuity necessary for development on a big program. This makes the project somewhat independent of individual staff members and therefore minimizes problems arising from changes in project staffing.

When the audit feature of HISTORIAN is used, all information about previous insertions and deletions and how they were caused can be recovered. Thus, for the above reasons, HISTORIAN was chosen as the automated code version tracking tool.

```
                                        •

                                        •

                                        •

                                        •

  585 continue                                                              tempest    840
      lskan(11)= lskan(10)+large                                            tempest    841
c**                                                                         tempest    842
c**   set array size and other index parameters                             tempest    843
c**                                                                         tempest    844
      nj        = nr-1                                                      tempest    845
      nk        = nz-1                                                      tempest    846
      ni        = nx-1                                                      tempest    847
      if(nj.eq.0) nj = 1                                                    tempest    848
      if(nk.eq.0) nk = 1                                                    tempest    849
      if(ni.eq.0) ni = 1                                                    tempest    850
                                        •

                                        •

                                        •

      ndc(nn)   = ndata(1)*npack2+ndata(2)*npack +ndata(3)                  input      387
      ndcmax    = max(ndcmax,ndr,ndz,ndx)                                   input      388
      lslpmx    = max(lslpmx,nsr,nsz,nsx)                                   input      389
  145 continue                                                              input      390
      go to 100                                                             input      391
c**   set up monitor indices                                 (kind = 5)     input      392
  150 continue                                                              input      393
      do 152 n = 1,10                                                       input      394
      mint(n)   = ndata(n)                                                  input      395
      if(mint(n).eq.0) mint(n) = 1                                          input      396
  152 continue                                                              input      397

                                        •

                                        •

                                        •

c**                                     •                                   momx       285
c**   compute density averages                                              momx       286
c**                                                                         momx       287
      rhojp1    = cvmgt(dp,rho(n+lwn),nt(n+lwn).eq.20.or.nt(n+lwn).gt.49)  momx       288
      rhokp1    = cvmgt(dp,rho(n+lnr),nt(n+lnr).eq.20.or.nt(n+lnr).gt.49)  momx       289
      rhoip1    = cvmgt(dp,rho(n+ljk),nt(n+ljk).eq.20.or.nt(n+ljk).gt.49)  momx       290
      rhor      = cvmgt(sr1*dp+sr2*rhojp1,dp0,couple)                       momx       291
      rhoz      = cvmgt(sz1*dp+sz2*rhokp1,dp0,couple)                       momx       292
      rhox      = cvmgt(sx1*dp+sx2*rhoip1,dp0,couple)                       momx       293
      smr(n)    = rhor*smr(n)                                               momx       294
      smz(n)    = rhoz*smz(n)                                               momx       295
      smx(n)    = rhox*smx(n)                                               momx       296
```

FIGURE C.3. HISTORIAN FORTRAN File Example

```
      usr(n)    = -smr(n)*up                                          momx      297
      vsr(n)    = -smz(n)*vp                                          momx      298
      wsr(n)    = -smx(n)*wp                                          momx      299
c**                                                                   momx      300
c**   compute advective momentum                                      momx      301
c**                                                                   momx      302
      duu       = sruu*(rc(j+1)*(abs(uap)*(up-ujp1)*sent +2.*uap*uap) momx      303
     1            +     rc(j)  *(abs(uam)*(up-ujm1)*sent -2.*uam*uam))momx      304
      dwu       = sxwu*(abs(sr2*wjp1 +sr1*wp)  *(up-uip1)*sent        momx      305
```

•

•

•

<u>FIGURE C.3</u>. (Contd)

```
*/     name = c_tempst_nb0.upd
*/     path = /800320/tempest/n29/a0
*/
*/     compatible with c_tempst_na0.xxx
*/          which is stored on the path: /800320/tempest/n29/a0
*/
*/     note that these updates create the "b1" version of tempest_n29
*/          from the "a0" version. there is no "b0" version as parent.
*/
*/
*/     ctss updates for the b1 version of tempest_n29
*/          these updates allow for debug output on file outd = tape14
*/          and the use of multiple titles
*/          these updates also create a new card in card group 1 called
*/          the dbug card. on this card the array idbug is loaded.later in
*/          the code it can be used for certain debug operations like debug
*/          writes to output or outd.
*/
*id dbfile0
*d tempest.34
     3   tape13,tape20,tape21,monsav,tape3=monsav,outd,tape14=outd,
*d vtype.2
      integer out,page,timunt,units,cname,outd,wrdsze,cf1,cf3
*i ncom.35
      common/ex7/outd
*i tempest.293
      outd     = 14
*i input.25
      outd = 14
*id lablfix
*i tempest.325
      ionce = 0
   38 continue
*i tempest.326
      if(ionce.ne.0) go to 42
*i tempest.329
   42 continue
*i tempest.328
      ionce = 1
*i tempest.345
      if(tlabel(20).eq.4h****) go to 38
*d tempest.331
*d tempest.338,345
*id dbswtch0
*i ncom.35
      common/ex8/idbug(200)
*id dbswtch1
*i tempest.362
      if(xcard.eq.4hdbug) go to 170
*i tempest.497
  170 continue
c**
```

FIGURE C.4. HISTORIAN Update File Example

```
c**   dbug array card
c**
      igroup = int(data(14))
      if(igroup.le.0) igroup = 1
      if(igroup.gt.20) igroup = 20
      istart = 10*(igroup-1) + 1
      l = istart - 1
      do 175 n = 1,10
```

FIGURE C.4. (Contd)

## DISTRIBUTION

No. of
Copies

OFFSITE

10 DOE Office of Scientific and
Technical Information

6 DOE Office of Civilian
Radioactive Waste Management
Forrestal Building
Washington, DC 20575
ATTN: L. H. Barrett, RW-33
C. R. Cooley, RW-40
J. R. Hilley, RW-30
S. H. Kale, RW-20
D. E Shejor, RW-32
R. Stein, RW-23

3 DOE Office of Defense Waste &
Transportation Management
GTN
Washington, DC 20545
ATTN: K. A. Chacey, DP-123
G. H. Daly, DP-124
T. B. Hindman, DP-12

3 DOE Office of Terminal Waste
Disposal & Remedial Action
GTN
Washington, DC 20545
ATTN: J. A. Coleman, NE-24
T. W. McIntosh, NE-24
W. R. Voight, NE-20
H. F. Walter, NE-24

A. T. Clark
Division of Fuel Material
Safety
Nuclear Regulatory Commission
Washington, DC 20555

V. Stello
Office of the Executive
Director for Operations
Mail Station 6209
Nuclear Regulatory Commission
Washington, DC 20555

No. of
Copies

S. Meyers
Environmental Protection Agency
Office of Radiation Programs
ANR-458
401 M Street, S.W.
Washington, DC 20460

J. M. McGough
DOE Albuquerque Operations
Office
P. O. Box 5400
Albuquerque, NM 87185

P. G. Hagan
Joint Intergration Office
Carlmont Executive 1
4308 Carlisle N.E.
Suite 101
Albuquerque, NM 87107

E. Maestas
DOE West Valley Demonstration
Project Office
P.O. Box 191
West Valley, NY 14171

3 DOE Idaho Operations Office
550 Second Street
Idaho Falls, ID 84301
ATTN: C. R. Enos
J. P. Hamric
S. T. Hinschberger

F. T. Fong
DOE San Francisco Operations
1333 Broadway
Oakland, CA 94612

M. R. Jugan
DOE Oak Ridge Operations Office
P.O. Box E
Oak Ridge, TN 37830

No. of
Copies

W. J. Goldston
DOE Savannah River Operations
Office
P.O. Box A
Aiken, SC 29801

M. J. Steidler
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

C. S. Abrams
Argonne National Laboratory
P.O. Box 2528
Idaho Falls, ID 83401

3 Battelle Memorial Institute
Project Management Division
505 King Avenue
Columbus, OH 43201
ATTN: W. A. Carbeiner
R. A. Nathan
Technical Library

L. D. Ramspott
Lawrence Livermore National
Laboratory
University of California
P.O. Box 808
Livermore, CA 94550

D. T. Oakley, MS 619
Los Alamos Scientific
Laboratory
P.O. Box 1663
Los Alamos, NM 87544

4 Oak Ridge National Laboratory
P.O. Box Y
Oak Ridge, TN 37830
ATTN: W. D. Burch
R. T. Jubin
L. J. Mezga
D. W. Turner

No. of
Copies

2 Sandia Laboratories
P.O. Box 5800
Albuquerque, NM 87185
ATTN: R. W. Lynch
Technical Library

J. R. Berreth
Westinghouse Idaho Nuclear
Co., Inc.
P. O. Box 4000
Idaho Falls, ID 83401

6 E. I. du Pont de Nemours
Company
Savannah River Laboratory
Aiken, SC 29801
ATTN: R. G. Baxter
M. D. Boersma
J. G. Glasscock
J. R. Knight
M. J. Plodinec
C. T. Randall

A. D. Rodgers
Mail Stop 2411
EG&G Idaho
P.O. Box 1625
Idaho Falls, ID 83415

R. Shaw
Electric Power Research
Institute
3412 Hillview Avenue
P.O. Box 10412
Palo Alto, CA 94304

4 West Valley Nuclear Service
Company
P.O. Box 191
West Valley, NY 14171
ATTN: J. C. Cwynar
J. E. Krauss
S. J. Marchette
J. M. Pope

No. of
Copies

J. L White, Chairman
Energy Research and Development
Authority
Empire State Plaza
Albany, NY 12223

ONSITE

9 DOE Richland Operations Office

E. A. Bracken
C. E. Collantes
C. R. DeLannoy
J. R. Hunter
O. L. Olson
J. M. Peterson
A. J. Rizzo
M. W. Shupe
J. J. Sutey

8 Westinghouse Hanford Company

J. M. Henderson
R. E. Lerch
H. E. McGuire
J. L. Scott
D. A. Turner
D. D. Wodrich
R. D. Wojtasek
B. A. Wolfe

50 Pacific Northwest Laboratory

C. R. Allen
W. W. Ballard, Jr.
W. F. Bonner
D. J. Bradley
R. A. Brouns (2)
M. J. Budden (5)
H. C. Burkholder
C. C. Chapman
T. T. Claudson
R. D. Dierks
M. R. Elmore
L. L. Eyler
D. W. Falleti (5)
R. K. Farnsworth (5)

No. of
Copies

R. W. Goles
W. O. Heath
D. K. Hilliard
D. E. Knowlton
M. R. Kreiter/J. E. Mendel
W. L. Kuhn
L. T. Lakey
D. E. Larson
J. L. McElroy
R. K. Nakaoka
J. M. Perez, Jr.
M. E. Peterson
W. A. Ross
D. H. Siemens
S. C. Slate
Publishing Coordination (2)
Technical Report Files (5)