1989

Small Sample Calorimetry, and Specific Heat of Gadolinium(1)barium(2-X)strontium(x)copper(3)oxygen(7).

Carter Harrison Watson

Louisiana State University and Agricultural & Mechanical College

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_disstheses

Recommended Citation

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Historical Dissertations and Theses by an authorized administrator of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.
INFORMATION TO USERS

The most advanced technology has been used to photograph and reproduce this manuscript from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book. These are also available as one exposure on a standard 35mm slide or as a 17" x 23" black and white photographic print for an additional charge.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700  800/521-0600
Small sample calorimetry, and specific heat of Gd$_{1}$Ba$_{2-x}$Sr$_{x}$Cu$_{3}$O$_{7}$

Watson, Carter Harrison, Ph.D.
The Louisiana State University and Agricultural and Mechanical Col., 1989
SMALL SAMPLE CALORIMETRY, AND
SPECIFIC HEAT OF
Gd$_{2}$Ba$_{2-x}$Sr$_{x}$Cu$_{3}$O$_{7}$

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Physics and Astronomy

by
Carter Harrison Watson
B.S. L.S.U. Shreveport, 1978
B.S. L.S.U. Shreveport, 1979
M.S. University of Mississippi, 1980
M.S.E.E. L.S.U. Baton Rouge, 1986
August 3, 1989
ACKNOWLEDGMENTS

I would like to thank all of the many people in and out of the Department of Physics and Astronomy that have been so supportive of my efforts here. I would like to give heartfelt thanks to several of these people.

To Naydene Watson, thanks for the long, long support. I wish daddy could have been here to see this.

To Prof. Roy G. Goodrich, thank you for your endless patience and support through the tortuous twists and turns of my stay here.

To Prof. Dana Browne, thank you for the enlightenment and selflessness of your instruction.

To Jing-chun Xu, thanks, I needed all of the help I could get.

Very special thanks to Dr. Ann Branch, who in many ways was the only person that could make this endeavor possible. Thank you for putting up with me and sticking with me even when I did not have the confidence necessary to finish.

To Bob and Aki Scott, its about time, is it not? Thanks.

To John McGurk and Scott McGoughan, thanks for putting up with all of my b.s. and mood swings. I really appreciated the support.

To Jim and Lynnette Pitts, and for a totally different reason, Lyna, thank you for the inspiration.

To Michel Reid and Jeff Tidwell, thanks for showing me what friends are.

To Prof. Paul Lee, thanks for the constant encouragement through the years.

To John & Sally & the Mitchells, thanks for being there.

Financial assistance for the preparation of this manuscript was provided by the Dr. Charles E. Coates Memorial Fund of the L.S.U. Foundation.
TABLE OF CONTENTS

ACKNOWLEDGEMENTS ii
TABLE OF CONTENTS iii
LIST OF FIGURES v
ABSTRACT ix
I. RECENT HISTORY 1
   1. THE RACE 1
   2. THE RARE-EARTHS 5
   3. L.S.U. HISTORY AND MOTIVATION 10
II. EXPERIMENTAL SECTION 16
   1. SAMPLE PREPARATION 16
   2. RELAXATION CALORIMETRY THEORY 17
      A. RELAXATION THEORY 17
      B. HEAT LEAK CALIBRATION CURVE 18
      C. RELAXATION CURVE 21
      D. EXTRACTION OF SPECIFIC HEAT 21
   3. AUTOMATED RELAXATION CALORIMETRY 25
      A. THE PROBE 25
      B. DATA COLLECTION HARDWARE 29
      C. DATA COLLECTION SOFTWARE 38
      D. THERMOMETER CALIBRATION 45
LIST OF FIGURES

Figure 1. Crystal structure of the Y$_1$Ba$_2$Cu$_3$O$_7$ class of high temperature superconductors, showing the placement of the barium, copper, yttrium, and oxygen atoms . . . . . 4

Figure 2. Anti-ferromagnetic ordering in Gd$_1$Ba$_2$Cu$_3$O$_7$, with moments aligned parallel and anti-parallel with the c-axis . . 11

Figure 3. Representation (not actual data) of a typical heat-leak calibration curve for the small sample calorimeter. Note the stepped structure, indicating steady state equilibrium temperatures for different applied power levels. Also note the block temperature curve at the bottom. . . . . . . . . 20

Figure 4. Temperature relaxation (decay) curve for a 30mg Gd$_{1.425}$Sr$_{0.575}$Cu$_3$O$_7$ sample in 0 applied magnetic field. Note the slight hump on the exponential decay part of the curve. This represents the magnetic ordering temperature. . . . . . . . . 22

Figure 5. Schematic representation of the small sample calorimeter probe, with more detailed representation of the $^3$He block. Note the radiation shield around the sample in the probe diagram on the left. . . . . . . . . 26
Probe pin-out showing the pin assignment map for the electrical feed-through at the top of the probe. 28

Figure 7. \(^3\)He/\(^4\)He plumbing diagram, showing the \(^3\)He injection system and recovery system. The circled T’s represent thermocouple gauges, and the circled arrows represent normal mechanical gauges. Note that the system can operate either as a \(^3\)He refrigerator, or as a \(^4\)He refrigerator. 30

Schematic representation of the electronic hardware layout of the system, showing the lock-in amplifier, the S.H.E. self-balancing bridge, the magnet and power supply, and the data collection sub-system consisting of an I.B.M. personal computer and a Tecmar labmaster data collection board. 32

Pin assignment map for the S.H.E. interface to the Tecmar labmaster board, showing the circuit board connections as well as the external port connections. 35

Condensed schematic of the S.H.E. digital I/O interface, showing the board level connections to the S.H.E. digital I/O bus. 37

Flow chart of the main data collector program. 39

Log(Resistance) versus Temperature for 1-Kohm chip resistor. 41

Deviation of predicted temperature from temperature for the previous 1-Kohm chip resistor. 43
Figure 14. Specific Heat divided by temperature versus temperature, for Gd$_3$Ba$_2$Sr$_x$Cu$_3$O$_y$ sample, with $x = 0.00$, and 0 applied magnetic field. 49

Figure 15. The best fits to the specific heat data of Brown (dots) using the model described in the text. The dashed line is for comparison ($J_x/I_x = 0.3$) to the best fit having $J_x/I_x = 70$ (solid curve). For both curves $J_x/I_x = 0.2$. 54

Figure 16. Specific Heat divided by temperature versus temperature, for the Gd$_3$Ba$_2$Sr$_x$Cu$_3$O$_y$ sample, with $x = 0.25$, and 0 applied magnetic field. 56

Figure 17. Specific Heat divided by temperature versus temperature, for Gd$_3$Ba$_2$Sr$_x$Cu$_3$O$_y$, with $x = 0.50$, and 0 applied magnetic field. 58

Figure 18. Two dimensional representation of two oppositely directed interpenetrating cubic ferromagnetic lattices. 61

Figure 19. Two dimensional representation of two tetragonal interpenetrating lattices, showing the interaction integrals for the nearest neighbors and the second nearest neighbors. 62

Figure 20. Plot of the Brillouin function $B_s(x)$ versus $x$ for $0 < x < 5$, and spin $s = 7/2$. 64

Figure 21. Plot of the specific heat divided by temperature versus temperature for Gd$_3$Ba$_2$Cu$_3$O$_7$ in applied magnetic
fields of 0, 0.5 T, 1.0 T, 1.5 T, and 2.0 Tesla.

The lower temperature peaks represent the Neel
temperature for the higher magnetic fields.. . . 68

Figure 22. Plot of the field dependent Neel temperature versus
magnetic field squared for Gd$_2$Ba$_2$Cu$_3$O$_{y}$.. . . . . 69

Figure 23. Table of calculated entropy changes for the specific
heat divided by temperature curves versus applied
magnetic field. . . . . . 70

Figure 24. Sensitivity function of the high temperature
calibrated germanium thermometer versus temperature. . . 82

Figure 25. Sensitivity function of the low temperature
calibrated germanium thermometer versus temperature. . . 85

Figure 26. Logarithm of the resistance of the high temperature germanium
thermometer versus temperature. . . . . . . . 87
ABSTRACT

The development of an automated small sample (0.5 to 50mg) calorimeter is discussed and evaluated. With this calorimeter, measurements of the specific heat of Gd$_2$Ba$_{2-x}$Sr$_x$Cu$_3$O$_y$ for $x = 0, 0.25, \text{ and } 0.5$ in the temperature range 0.6 to 3 K are reported. Measurements on the $x = 0$ sample agree with data collected by other groups. In addition, measurements in magnetic fields from 0 to 3 T for the $x = 0$ sample are reported. In each case a peak in the specific heat at the antiferromagnetic ordering temperature (the Neel temperature) is recorded and analyzed. Strontium doping does not appreciably change the Neel temperature, but drastically changes the shape of the specific heat curve below the ordering temperature. The Neel temperature is reduced proportional to the square of the applied magnetic field, becoming no longer observable for fields greater than 3.0 T. Mean field theory is applied to the Neel temperature shift in order to gain a qualitative description for this behavior. The model calculation of the spin wave contribution to the low temperature specific heat discussed in the paper indicates that strong couplings along the c axis and weak dipolar couplings in the a-b plane are responsible for the antiferromagnetic order. The strontium doping further substantiates this argument by showing that the removal of the barium ions weakens the gadolinium coupling in the c direction by removing the super-exchange path.
CHAPTER I
RECENT HISTORY OF SUPERCONDUCTIVITY

1. The Race

Since the discovery of superconductivity in 1911 by Onnes, the superconducting transition temperatures $T_c$ of known materials have risen very slowly from 3 K for Hg discovered in 1911, to 23.2 K for Nb$_2$Ge, discovered in 1973. That slow rate of discovery of materials with higher $T_c$ was shattered in January 1986 by Bednorz and Muller at the I.B.M. Research Laboratory in Switzerland when they discovered superconductivity at $T_c = 30$ K in the La-Ba-Cu-O perovskite system developed by Michel and Raveau. After several attempts to find a high $T_c$ in the La-Ni-O system, the Bednorz and Muller group tried the La-Ba-Cu-O perovskite system. This system showed a drop in resistivity at $T_c = 30$ K.

The importance of this discovery was rated so high that the group published without performing magnetic measurements to show the presence of the Meissner effect within the system. This breakthrough was further substantiated when Chu et al. reproduced the high $T_c$ in the same system and showed that by using a non-coprecipitation solid diffusion technique, $T_c$ could be increased to 35 K in the same system.

Soon after publication of the Bednorz and Muller paper, the magnetic measurements were performed and the transition from paramagnetism to
diamagnetism (the Meissner effect) was shown to be related to the resistivity measurements.\(^7\) This result indicated that the 23.2 K barrier for superconductivity had indeed been broken. Further x-ray analysis showed BaLa\(_2\)CuO\(_4\) to be the superconducting phase of the compound. Addition of alkaline earth elements Sr and Ca to the La-based compound pushed \(T_c\) to 40 K.

Cava et al.\(^8\) showed that the \(T_c\) of La\(_{1,8}\)Sr\(_{0,2}\)CuO\(_4\) was sensitive to annealing methods. For example, samples annealed in air were composed of a mixture of metallic and insulating regions with \(T_c = 36.5\) K and a resistive transition width from normal to superconducting of 5.3 K. Samples annealed in oxygen had metallic properties with \(T_c = 38.5\) K and resistive transition widths of 1.4 K. This indicates that sample preparation plays a critical role in the physical properties of these superconductors.

Some of the early work on La\(_{1,8}\)Sr\(_{0,2}\)CuO\(_4\) was directed toward comparing properties of this material to the theory of superconductivity developed by Bardeen, Cooper, and Schrieffer (B.C.S.)\(^9\). The energy gap, a measure of the energy needed to break Cooper pairs in a B.C.S. superconductor, of La\(_{1,8}\)Sr\(_{0,2}\)CuO\(_4\) was studied by Walter et al.\(^10\) using far-infrared reflectance measurements. From their data they arrived at an energy gap of 60 cm\(^{-1}\), which is smaller than the B.C.S. prediction for the energy gap at \(T_c = 36\) K, 87 cm\(^{-1}\), by approximately 50%. Nuclear magnetic resonance experiments were performed to show the existence of an energy gap with a ratio \(2\Delta/k_B T = 7.1\), indicating a strong coupling present in the La-Ba-Cu-O system. However, infrared reflectance data taken on powdered samples oriented along the c-axis showed a ratio of 2.5, indicating that the coupling between planes
was weak. These early experimental results are conflicting and probably are due to a large anisotropy not normally found in other superconductors.

Chu et al. studied the effects of pressure on the La-Ba-Cu-O system and found that the application of pressure enhanced $T_c$ at a rate of $10^3$ kbar, and subsequently raised $T_c$ to 57 K. This result suggested that substituting a smaller ionic radii element for La, and thereby reducing the lattice spacing, might raise $T_c$ even higher. This substitution was done, and $T_c$ increased to 92 K in February 1987 through a joint effort between the University of Alabama at Huntsville and the University of Houston. The superconducting phase responsible for this incredibly high transition temperature was subsequently isolated by Cava et al. as Y$_2$Ba$_2$Cu$_3$O$_7$. The accepted structure of this compound is shown in figure 1.

Following the discovery of the 90 K $T_c$ in Y$_2$Ba$_2$Cu$_3$O$_7$ in the spring of 1987 over 1000 papers have been published on this and related materials. In addition, two new high $T_c$ materials, Bi$_2$Sr$_2$CaCu$_2$O$_8$ with a $T_c$ of 110 K and Tl$_2$Ba$_2$Ca$_2$Cu$_2$O$_8$ with a $T_c$ of 120 K, have been synthesized. A complete theory explaining the superconductivity in all of these compounds has yet to be achieved.

Prior to the discovery of these high $T_c$ materials, one of the sacred rules was that the introduction of impurity atoms with localized magnetic moments into the material would quickly destroy the superconductivity. In fact, the distinction between magnetic and superconducting interactions in superconductors has been a subject of continuing interest. Very early in 1987 it was found that in the Y$_2$Ba$_2$Cu$_3$O$_7$ material, not only could magnetic impurities be inserted at the yttrium sites without destroying superconductivity, but all of the yttrium atoms could be
Figure 1
replaced with atoms having localized magnetic moments with little effect on the superconductivity.

2. The Rare-Earths.

Since the discovery of superconductivity in the $Y_1Ba_2Cu_3O_7$ system, one avenue of research aimed at trying to determine the nature of the superconductivity in these compounds has been the effect of substituents at various atomic sites in the lattice. Willis et al.\textsuperscript{17} at Los Alamos National Laboratory reported little change in $T_c$ for six rare earth systems, $REBa_2Cu_3O_7$ (where $RE = Sm, Eu, Gd, Tb, Dy, Ho$), having a structure identical to $Y_1Ba_2Cu_3O_7$. However, no superconductivity was found in the compounds with Ce, Pr, or Pm. Each of these rare earth atoms has localized unpaired f electrons, which gives the $RE^{3+}$ configuration a large magnetic moment. This result led to the speculation that the magnetic 4f ion is ineffective at depressing $T_c$, and that superconductivity must be held within the Cu-O layers.\textsuperscript{18} The Los Alamos group\textsuperscript{19} performed resistivity, magnetic susceptibility, and heat capacity measurements on these systems. For the Gd system, the reported $T_c$ was 95 K, and bulk superconductivity indicated by a Meissner effect was observed.

In other metallic systems, Gd depresses superconductivity because the magnetic moment due to its 7 unpaired 4f electrons have a tendency to break Cooper pairs. However, in the $Gd,Ba_2Cu_3O_7$ system the effect of the Gd$^{3+}$ moments must be negligible because the $T_c$ of the Gd based system is nearly the same as that of the yttrium based system, which has no unpaired f electrons.
D. C. magnetic susceptibility measurements were performed at Oak Ridge National Laboratory on the Gd system in order to study the interaction of these 4f electrons on the superconductivity of the system. The magnetic susceptibility measurements yielded an effective moment of $\mu_{\text{eff}} = 7.84\mu_B$, which agrees well with the value of $7.94\mu_B$ predicted by Hund’s rules for the $^8S_{7/2}$ Gd$^{3+}$ ion. Also, the Curie-Weiss constant comes out to be +4.3 K, corresponding to a net antiferromagnetic interaction between the Gd moments.

Specific heat data taken above 15 K show a Debye-like behavior (i.e., when examined as $C_p/T$ vs. $T^2$, there is a linear behavior above 15 K, that corresponds to the Debye $T^3$ lattice contribution to the specific heat). Below 10 K, however, there is a marked rise in $C_p$ that is attributed to the higher temperature contribution of the antiferromagnetic ordering peak that occurs at 2.24 K. Along with the specific heat data, the resistivity, and the magnetic susceptibility of the Los Alamos data agrees closely with those of other workers.

Tang et al. reported the coexistence of magnetic ordering and superconductivity in the Gd$_3$Ba$_2$(Cu$_{0.94}$Fe$_{0.06}$)$_3$O$_{9.5}$ system at 4.2 K. Resistivity measurements showed a drop in resistance at $T_c = 42$ K. The 6% substitution for Cu causes scattering of unpaired electrons by the large Fe moment, which suppresses superconductivity. Mossbauer spectra of the same Fe-doped sample taken at 4.2 K shows evidence that the Fe moments are magnetically ordered at 4.2 K. This allows the possibility of coexistence of magnetic ordering and superconductivity in the Cu-O layers or in the CuO-Ba-CuO sublattices in contrast with the magnetic ions and the superconducting electrons occupying separate...
sublattices. However, their paper did not clarify the placement of the Fe sites, so the iron ions may not be in the same sublattice.

As expected, the magnetic ordering among the Fe$^{3+}$ moments does not depend on the existence of Gd$^{3+}$, since the same phenomenon occurred in the Y$_{1}$Ba$_{2}$(Cu$_{0.5}$Fe$_{0.5}$)$_{3}$O$_{9.4}$ system (yttrium has no magnetic moment). Thus, the magnetic ordering among the Fe moments must be confined to the CuO-Ba-CuO sublattices or the CuO layers.

Chien et al.\textsuperscript{26} showed that superconductivity is completely suppressed if only 10\% of the copper is replaced with zinc. This further supports the premise that the superconductivity occurs in the CuO-Ba-CuO sublattices or the CuO layers, rather than in the rare earth sublattice.

Pressure studies on the RE$_{1}$Ba$_{2}$Cu$_{3}$O$_{7}$ compounds were also performed.\textsuperscript{27} In the study made by Hor et al. on the Y$_{1}$Ba$_{2}$Cu$_{3}$O$_{7}$ system, it was found that T$_{c}$ increased only weakly with an increase in pressure. This questionable result was due to the masking of the pressure dependence of the system by the presence of additional impurity phases within the system. In the study made by Borges\textsuperscript{23} on more pure systems, an increase in T$_{c}$ with an increase in pressure was shown. The rate of increase in T$_{c}$ due to pressure is indicated below:
<table>
<thead>
<tr>
<th>Element</th>
<th>dTc/dP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yttrium</td>
<td>0.09 K/Kbar</td>
</tr>
<tr>
<td>Gd</td>
<td>0.13 K/Kbar</td>
</tr>
<tr>
<td>Er</td>
<td>0.14 K/Kbar</td>
</tr>
<tr>
<td>Yb</td>
<td>0.19 K/Kbar</td>
</tr>
</tbody>
</table>

This indicates that dTc/dP increases with an increase in atomic weight of the rare earth ion. This implies that the thermal expansion change from the normal state to the superconducting state is positive. The similar behavior in all four compounds indicates that the rare earth ions local moment has no affect on the pressure response.

Torrance et al. have described the structure of the RE$_2$Ba$_2$Cu$_3$O$_7$ compounds in terms of a primary framework of stacked Cu-O$_6$ octahedra and a secondary structure of ordered oxygen vacancies. The primary framework is determined by the ratio of noncopper metal atoms to copper atoms. The secondary structure may take the form of channels or sheets of vacancies or both. These two features determine the copper valence; that is, the Cu$^{2+}$ compounds are semiconductors, while the mixed valence compounds are conductors.

The Y$_1$Ba$_2$Cu$_3$O$_{7-x}$ compound was shown to exist in two phases: the tetragonal phase, Y$_1$Ba$_2$Cu$_3$O$_6$, and the orthorhombic phase, Y$_1$Ba$_2$Cu$_3$O$_7$. Crystal structures of the RE$_2$Ba$_2$Cu$_3$O$_{7-x}$ compounds were found to also come in the two phases of similar structure.

The physical properties of these two phases were studied by Dunlap et al. The orthorhombic phase is made into the tetragonal phase by annealing it in a
nitrogen atmosphere. This change in structure is due to a reordering of the oxygen occupancies, while leaving the rare earth sublattice unchanged. The orthorhombic phase shows a $T_c = 95$ K, in agreement with other work, but the tetragonal phase is semi-conducting at low temperatures with no superconductivity present.

The heat capacity data shows the Neel temperature occurs at 2.24 K for both phases. The shoulder first reported by Brown et al. in the superconducting phase is also seen in the tetragonal phase. Thus, the magnetic characteristics of the system are independent of the conduction characteristics. It therefore follows that the rare earth ions are isolated from the rest of the system, existing as a sublattice that may interact through dipolar interactions, causing the magnetic ordering that occurs within these compounds (Dipole-dipole energies are on the order of 1 K, in agreement with the Neel temperature of 2.24 K). This ionic isolation is probably the reason that the magnetic rare earth ions may be substituted into the compounds without disrupting their superconductivity, since the conduction electrons are not affected by the pair breaking effect of the magnetic ions. This is in agreement with band structure calculations for the $\text{Y}_1\text{Ba}_2\text{Cu}_3\text{O}_7$ system which shows a low conduction electron density at the yttrium sites.

Single crystal x-ray measurements have been performed on both phases of $\text{Y}_1\text{Ba}_2\text{Cu}_3\text{O}_7$. The lattice constants for the orthorhombic phase are:

$$a = 3.819 \, \text{Å} \quad b = 3.891 \, \text{Å} \quad c = 11.746 \, \text{Å}$$

For the tetragonal phase, the constants are:

$$a = 3.851 \, \text{Å} \quad b = 3.851 \, \text{Å} \quad c = 11.857 \, \text{Å}$$
In the orthorhombic phase, the oxygens order along the b axis with a fractional occupancy of 1. The Cu-O$_{4z}$ units are square planar. In the tetragonal phase, the fractional occupancy is 0.25 along the a and b axes, forming a disordered two dimensional Cu-O unit.

In a similar vein, the magnetic ordering of the gadolinium sublattice in the superconducting phase (powder) was examined from the observed magnetic neutron Bragg reflections (neutrons at 3.006 Å) of Paul et al. Derived lattice constants for the orthorhombic phase were

\[
a = 3.844 \text{ Å} \quad b = 3.905 \text{ Å} \quad c = 11.714 \text{ Å } +/- 0.005
\]

These reflections show a Gd$^{3+}$ ordering that is face centered, orthorhombic, on the up-spin lattice with all nearest Gd neighbors antiferromagnetically aligned. This result shows that the diffraction pattern doubles the unit cell in all three directions; therefore, the magnetic structure is three-dimensional AFM ordering at 2.24 K. The magnetic structure is shown in Figure 2.

The above experiments have indicated that the Gd$_3$Ba$_2$Cu$_4$O$_7$ structure is identical to that of the Y$_3$Ba$_2$Cu$_4$O$_7$ structure. Many of the non-magnetic properties of these two compounds are the same. However, the Gd$_3$Ba$_2$Cu$_4$O$_7$ compound demonstrates the existence of magnetic order and superconductivity. The compound has three dimensional antiferromagnetic ordering, which must be coupling through the superconducting planes and chains. Further study of this system is indicated.


The efforts on high temperature superconductivity at L.S.U. began in March 1987 with initial attempts to produce the high $T_c$ superconductors. Considerable
effort was expended in getting the sample preparations correct, since the journal articles at that time were sketchy (at best) on their sample preparation techniques. As it turned out, the grinding and annealing procedures were critical to producing a usable sample. By the summer of 1987, superconducting samples were being routinely produced. Of course, some of the best ceramic insulators were also being produced. Resistivity experiments, R. F. surface impedance measurements, and various other experiments were started at that time.

One of the many probes for determining the physics of solids is the measurement of the temperature-dependent specific heat. The measurement of the specific heat can yield detailed information about the different physical mechanisms and processes affecting the properties of solids, liquids, and gases. Using specific heat data, ordering transitions, superconducting transitions, spin-waves, and many other phenomena can be probed.

During the summer of 1987, this author was involved with the development of an automated small ( < 50mg) sample calorimeter, to be used primarily for making specific heat measurements on a single crystal, in the temperature range of 0.5 to 20 K. This range is obviously well below the temperature ranges necessary to make measurements near the transition temperature on most of the high Tc compounds. However, one of these compounds, Gd1 Ba2 Cu3 O7, was especially interesting since it demonstrated the coexistence of a local magnetic order and superconductivity. It just so happened that this compound had an anomalous specific heat transition at 2.2 K, right in the mid-range of the small sample calorimeter.
An early model developed here at L.S.U., based on specific heat data, indicated that it may be possible for the Gd$_3$Ba$_2$Cu$_3$O$_7$ lattice to order with both ferromagnetic and (total) antiferromagnetic ordering. A paper was submitted, and later withdrawn when the magnetic diffraction data of Paul et al. became available. This spurred an intense effort to understand the physics of this low-temperature transition, and a more detailed program of study to measure the specific heat of Gd$_3$Ba$_2$Cu$_3$O$_7$ was instigated. This involved the measurement of the specific heat in magnetic fields from 0 T to 5 T, and temperatures from 0.5 K to 5 K. The reason that the magnetic field dependent specific heat was measured was to confirm or deny the reports that the peak at 2.2 K was in fact an antiferromagnetic (three dimensional) ordering transition.

The need arose later in the program to probe the c-axis magnetic coupling of the Gd ions in the three dimensional lattice. In order to determine if the barium ions were involved with this coupling, a new program of strontium substitution for the barium ions was started. Previous work had shown that this was possible in the Y$_1$Ba$_2$Cu$_3$O$_7$ system. Since strontium does not have the same electronic orbital overlaps as those of barium, this was expected to make a difference in the specific heat, if the barium ions had a significant role to play in the c-axis coupling of the gadolinium ions. If the barium ions do not play a significant role, then the specific heat should remain relatively unchanged, since the bulk of the specific heat curve is involved with the magnetic ordering transition.

Initial strontium substitution levels were 0 (pure barium), 0.25 (7 barium ions per strontium ion), and 0.50 (3 bariums per strontium ion). This was the highest
strontium concentration (in 0.25 steps) that allowed the sample to remain superconducting. Although this study was not directly investigating the superconducting mechanisms and properties of the Gd-123 samples, we chose the 0.50 strontium substitution as an upper limit on the replacement study in order to assure that the crystallites maintained the 123 structure and were otherwise no different than the pure barium, superconducting Gd-123 sample.

In the chapters that follow, a detailed discussion of the small sample calorimeter will be undertaken. This will include an analysis of the system accuracy, developed in the appendices. In addition, the development and the results of the two investigations that were discussed above will be examined in detail. Finally, a complete listing of the data collection and data analysis programs will be presented.

Admittedly, the Gd$_3$Ba$_2$Cu$_3$O$_7$ compound can be prepared in large amounts, so a small sample calorimeter is not technically necessary for pressed-pellet specific heat measurements. However, during 1987, the small sample calorimeter was the only low-temperature specific heat apparatus that was operational in the laboratory. Also, since the specific heats of magnetic materials are normally much larger than the specific heats of non-magnetic materials, the thermal time constants involved in any calorimetry experiment involving these materials will be very large. Thus it would be preferable to use smaller samples, if possible, in order to reduce the times over which the experiments must be run. This is desirable since the time between liquid $^4$He transfers is a strongly governing factor in the length of specific heat
measurement experiments at low temperatures, and the price of liquid helium is high. Typical helium transfer intervals for the experiment bays in the low temperature laboratory are on the order of 6 hours. Typical data collection runs for the small sample calorimeter, on a 30mg Gd$_3$Ba$_2$Cu$_3$O$_7$ sample, are on the order of 4 hours. Therefore, significantly larger samples would have to have a higher heat-leak in order to reduce the temperature relaxation time. This is not always realizable, since the heat-leaks of the sample and block determine the length of time that the $^3$He refrigerator can be run.

Also, one of the ultimate goals of the sample preparation experiments at L.S.U. is the production of single crystals of the high $T_c$ compounds. These crystals will most likely have masses on the order of less than a milligram, necessitating the use of a small sample calorimeter. With the calorimetry system already developed and functional, experimental lag time will be significantly reduced.
CHAPTER II

EXPERIMENTAL SECTION

1. SAMPLE PREPARATION

The superconducting material, Gd$_2$Ba$_2$Cu$_3$O$_{7-}$, used in the experiment was prepared by mixing and grinding suitable quantities of Gd$_2$O$_3$, CuO, and BaCO$_3$ in order to satisfy the following equation

$$
Gd_2O_3 + 4BaCO_3 + 6CuO + 0.5O_2 \Rightarrow 2Gd_2Ba_2Cu_3O_7 + 4CO_2
$$

The mixture was placed in a ceramic boat and heated to 960 °C in O$_2$ for 8 hours. The reacted powder was then removed from the furnace, allowed to cool to room temperature, then reground to a very fine powder of almost buttery consistency, and heated again to 960 °C. Pressed pellets were made from this powder by placing the powder in an hydraulic press and applying approximately 40 tons per square inch pressure. This pressure was maintained for several minutes while the pellet relaxed. Pressure was reapplied until the sample no longer relaxed. The resulting pellets were typically 0.5 inches diameter by 0.02 to 0.25 inches thick. The pellets were then sintered at 960 °C for 16 hours, cooled to 475 °C at a rate of 0.5 °C/min, and soaked at 475 °C for 12 hours all in 1 atmosphere of flowing O$_2$. They were then allowed to cool to room temperature. A powder X-ray diffraction pattern confirmed the ortho-rhombic 123 structure, and AC susceptibility measurements confirmed superconductivity, with a measured transition temperature above 90K.
The $\text{Gd}_2\text{Ba}_{2x}\text{Sr}_x\text{Cu}_3\text{O}_7$ samples were prepared in a similar manner, except that the $\text{BaCO}_3$ was replaced by $[(1-x/2)\text{BaCO}_3 + (x/2)\text{SrCO}_3]$ (in formula-weight units). Two of these partial strontium substitution samples were prepared, with $x$ values of 0.5 and 0.25. Powder X-ray diffraction again confirmed the 123 structure, with no impurity phase present. The transition temperatures for the strontium doped samples were measured by AC susceptibility to be 80 K for the $x = 0.25$ sample, and 50 K for the $x = 0.5$ sample.

2. RELAXATION CALORIMETRY -- THEORY

a. Relaxation Calorimetry -- Theory

There are essentially four standard methods for experimentally determining the specific heat of a material: adiabatic, continuous warming, a.c. calorimetry, and thermal relaxation calorimetry. They all have distinct advantages and disadvantages. Adiabatic calorimetry is the simplest to understand, since the method involves heating an adiabatically shielded sample with a controlled power input and measuring the temperature rise. Unfortunately, it usually requires large sample masses (since the thermometers and the wires leading to them must be accounted for in the specific heat calculation, as well any interference caused by the suspension system), careful attention to shielding, and gives only discretely spaced specific heat information. A. C. calorimetry works quite well for small samples, but requires careful attention to the thermal time constants of the system, and can easily become unusable for samples with high internal time constants. In the continuous warming method, heat at a constant rate $P$ is added to the sample and the resulting
rate of increase of temperature is measured. However, it does require accurate information on the sample heater and excellent thermal shielding of the sample. The method used throughout this paper was relaxation calorimetry, which can provide a continuous readout of the specific heat of small samples (typically < 100mg) but requires an accurate calibration of system heat leaks.

The basic concept of thermal relaxation is as follows: a sample and thermometer are thermally attached to a heat reservoir at temperature $T_0$; a constant power input $P$ is applied to the sample, raising the temperature of the sample to some value $T$ (after achieving steady-state equilibrium); the power $P$ is then shut off and the continuous temperature decay of the sample is recorded.

b. The Heat-Leak Calibration curve

Figure 3 gives an example of a typical heat-link heat-leak calibration curve. The main points of interest of this curve are the zero-slope steps where the bolometer/sample/leads conglomerate has reached a steady-state equilibrium. At each of these points a fixed amount of D.C. power is being applied to the sample heater through the gold-copper leads attaching the bolometer to a heat reservoir (the $^3$He block, described later). The sample and bolometer then reaches thermal steady-state equilibrium through the heat leak provided by the gold-copper leads mentioned previously. The other main point of interest is the temperature of the block itself is also shown on the actual data collection display, usually for debugging purposes. This is indicative of when the block temperature has stabilized via its electronic controller.
Each step of the heat-calibration curve corresponds to a different power level being dissipated in the sample heater. Since the system is in a steady-state equilibrium, the electrical power being supplied to the heater is balanced by all of the heat-losses in the system. Convection is not a significant factor in the heat loss equation for this system since $^4$He pressures in the vacuum can are below $10^{-6}$ torr, and are therefore ignored in the model used for the curve fit for the calibration. Radiation losses are also insignificant since the sample has a radiation shield between it and the 1.2K $^4$He bath, and the temperature difference between the sample and the thermal shield typically does not exceed 3K. This can be readily verified by examining the Stefan-Boltzmann law for energy emitted/absorbed by a not-so-black body

$$e(T) = \alpha \sigma_b A (T_1^4 - T_2^4),$$

where Stefan's constant $\sigma_b = 5.67 \times 10^{-12}$ Watts cm$^{-2}$ K$^{-4}$, $A$ is the cross sectional area of the body, $\alpha$ is the absorptivity of the body (on the order of 0.6 for oxidized Cu), and $T_2$ and $T_1$ are the temperatures of the surrounding shield and the body, respectively. Using the above values, with $T_2 = 0.5$ K, and $T_1 = 4.2$ K, gives a total power of approximately $10^{-9}$ Watts. This is of the same order as the measurement power, which is several orders of magnitude less than the heating power; therefore, it is neglected. This leaves conduction as the most important heat transfer mechanism. If one knows the electrical power loss versus the temperature rise of the system, one has an empirical calibration of the heat-link heat-loss of the system. With the use of a simple model for the thermal conductance of the wire,
Figure 3
one need not have a large number of steps for this calibration curve fit.  

\textit{c. The Relaxation Curve}

A relaxation curve is shown in Figure 4 and is typical for a sample with a transition. The main features of interest are the initial sudden drop from the last heat step, the slowing down of the relaxation curve near the transition, the final exponential approach to the block temperature, and the display of the block temperature.

The curves are normally acquired over a period of hours, since the decay time constants can easily be as large as thousands of seconds. The temperature differences between the base temperature and the sample temperature are usually held to less than several degrees. This is done in order to minimize the number of steps necessary for the heat-link calibration curve, as well as to minimize temperature gradients in the conglomerate, and to avoid the need for radiative loss corrections.

\textit{d. Extraction of Specific Heat}

Once the above curves have been acquired, it is possible to extract the specific heat of the sample over the temperature range of the decay.
Figure 4
This is accomplished in a series of steps:

i) calculate the heat-link heat-loss as a function of temperature

ii) perform a smoothing-fit of the decay curve

iii) calculate the heat loss and the time derivative of the temperature

iv) calculate the specific heat

The first step is normally accomplished by performing a simple linear least-squares of strain-gauge resistive heating versus temperature squared, i.e. fitting power loss (equal to resistive heating under steady state conditions) via the equation:

$$P_l(T;T_0) = K(T)AT^2 - T_0^2$$

where $K$ is the thermal conductivity, $A$ is the cross-sectional area of the wire, and $T_0$ is the base temperature of the block.

This originally empirical result can be verified by the following argument: consider a one-dimensional heat conductor of length $L$ connected between two heat reservoirs at temperatures $T_0$ and $T$. Fourier's Law states that

$$\dot{q} = -KA\nabla T,$$

which will reduce in one-dimensional systems to

$$\frac{dq}{dt} = -kA\frac{dT}{dx}.$$

If we integrate both sides of the equation with respect to $x$ we arrive at

$$\int_0^L (dq/dt)dx = \int_0^L K(T)A(dT/dx)dx.$$
For a steady-state equilibrium condition we know that \( dq/dt \) is constant along the heat-conductor, so it can be pulled out of the integral. This reduces the left hand side to the result of \( (dq/dt) \cdot \text{Length} \). So the equation is reduced to:

\[
L \frac{dq}{dt} = -A \int_{T_0}^{T} K(T) dT .
\]

Assuming that the thermal conductivity \( K(T) \) is simply proportional to temperature,\(^4\), since it is a metallic alloy,

\[ K(T) = kT , \]

then one can integrate the right hand side of our original equation to give

\[
\frac{dq}{dt} = -kC(T^2 - T_0^2) .
\]

Smoothing of the decay curve, step ii), is normally accomplished by either performing a smoothing cubic spline calculation, performing a simple average of neighboring points, or by performing a local least-squares (LS) calculation.

Step iii) is performed by using the results of parts i) and ii) : at each point on the decay curve, calculate the smoothed temperature, and the smoothed derivative (the derivative comes directly out of the smoothing spline calculation, if this is the method of smoothing used; there is no need to do additional calculations), and from the smoothed temperature calculate the power loss down the leads to the \(^3\)He block.

By using the above information, one can calculate the heat capacity by:

\[
C_p(T) = - P_{L}(T,T_0) / (dT/dt)
\]

which yields the heat capacity of the sample over the temperature of the decay. By scaling, using the sample mass and molecular weight of the sample, one gets the specific heat of the material via
\[ c_p = (\text{Mol. Weight}) \cdot C_p / M \] 

3. AUTOMATED RELAXATION CALORIMETRY

a. The probe

Figure 5 shows a diagram of the probe used for small sample calorimetry. The sample, usually of mass less than 100mg, is sandwiched between the sample thermometer and the sample heater. The sample heater is a 120-ohm metallic strain gauge. The strain gauge makes an excellent heater, since the resistance of the gauge is almost constant over the range of use (0.5K to 25K), its mass is small (typically < 3mg), its resistance is stable with respect to cryogenic cycling, and it has a small specific heat in the temperature ranges of interest.\(^4^6\) The sample thermometer is typically a Bismuth-Ruthenate 1k-ohm, 10k-ohm, or 100k-ohm Dale chip resistor that has been partially mass-shaved. The resistors have small mass (< 3mg), good temperature cycling properties, a relatively simple functional dependence on temperature, and a small magneto-resistance.\(^4^7\) The heater and thermometer are attached on opposite sides of the sample with G.E. 7031 varnish. The sample and addenda then have gold-copper leads (typically 0.005cm diameter by 2cm long, 93% Au - 7% Cu) attached to them with low melting point solder. These leads are then attached to copper pads (four pads in all) that are mechanically attached to the bottom of the \(^3\)He pot. The four copper pads are insulated from the pot by a layer of varnish impregnated with CuO, the latter acting strictly as an electrically
Figure 5
insulating, thermally conductive standoff. The resistance of the thermometer and heater is measured by four-wire measurements at the pads (i.e. the lead resistances from the pads to the sample thermometer and heater are not directly measured during the experiment, but this is typically less than 0.5 ohms). The wires coming from the sample thermometer/heater are four conductor QL-36 wire sold by Lakeshore Cryogenics. The wire consists of four 36 gauge individually insulated phosphor-bronze wires. They are thermally anchored at the pot with GE 7031 varnish, and at several places along the 3He pumping line with Apeozan grease, and again at the vacuum can flange with varnish.

Also attached to the block are several block thermometers. First, there is a 100 k-ohm chip resistor, read by a Keithley 179A D.V.M. in a two-wire mode, that gives an indication of the temperature of the block in the range of 5K to 50K. Second, there is the 1k-ohm or 5k-ohm chip resistor which is used for the 0.5K to 5K range, and also allows usage in a magnetic field. Third, are the calibrated Germanium resistors, provided by Lakeshore Cryogenics (GR-200A-10, GR-200A-1000) used for calibrating the above resistors in the ranges 0.5K to 2K, and from 1.5K to 25K. The second and third resistors are measured with a S.H.E. self-balancing conductance bridge that normally provides a 100 micro-volt excitation, providing a measurement power of much less than $10^{-9}$ watts. The leads leaving all three of these resistors are of the same type as the four conductor leads mentioned above, and are thermally anchored the same way.

The sample, sample heater, sample thermometer, and all of the block thermometers are enclosed inside a copper foil can within the vacuum can. This
acts as a radiation shield to keep the sample from "seeing" the heat radiated by the vacuum can, and gaining or losing heat by radiation exchange with the vacuum can, that is normally at 1.3K during the course of a normal run. The radiation shield is kept at the same temperature as the $^3$He pot, which is temperature controlled.

The block temperature is changed by applying 0-10 volts to a 150 ohm heater wire wound around the $^3$He pot. The pot heater is connected (two-wire) to a Harrison 70 volt programmable power supply that is controlled by the computer.

**Probe Pinout**

<table>
<thead>
<tr>
<th>Pin #</th>
<th>Function</th>
<th>Pin #</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Block 1k chip V+</td>
<td>14</td>
<td>Strain gauge I+</td>
</tr>
<tr>
<td>2</td>
<td>Block 1k chip I+</td>
<td>15</td>
<td>Strain gauge I-</td>
</tr>
<tr>
<td>3</td>
<td>Block 1k chip I-</td>
<td>16</td>
<td>Strain gauge V-</td>
</tr>
<tr>
<td>4</td>
<td>Block 1k chip V-</td>
<td>17</td>
<td>Block Germ I-</td>
</tr>
<tr>
<td>5</td>
<td>Bol 1k chip V-</td>
<td>18</td>
<td>Block Germ I+</td>
</tr>
<tr>
<td>6</td>
<td>Bol 1k chip I-</td>
<td>19</td>
<td>Block Germ V+</td>
</tr>
<tr>
<td>7</td>
<td>Bol 1k chip V+</td>
<td>20</td>
<td>Block Germ V-</td>
</tr>
<tr>
<td>8</td>
<td>Bol 1k chip I+</td>
<td>21</td>
<td>unused</td>
</tr>
<tr>
<td>9</td>
<td>Block 100k chip R1</td>
<td>22</td>
<td>unused</td>
</tr>
<tr>
<td>10</td>
<td>Block 100k chip R2</td>
<td>23</td>
<td>unused</td>
</tr>
<tr>
<td>11</td>
<td>Pot heater V+</td>
<td>24</td>
<td>unused</td>
</tr>
<tr>
<td>12</td>
<td>Pot heater V-</td>
<td>25</td>
<td>unused</td>
</tr>
<tr>
<td>13</td>
<td>Strain gauge V+</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6**

Above the $^3$He pot is the $^3$He pumping line that allows the probe to achieve temperatures below 0.6K. This line is of large diameter in order to maximize the gas conductance through the line. Also attached to the outside of the above line are a number of light baffles. These thin stainless steel disks shield the vacuum can and $^3$He pumping line and pot from the 300K room temperature photons. The $^3$He is recovered from the pot via the Alcatel mechanical pump, and the output from this
pump is put into another pump for storage in a separate tank, thus allowing the system to be used in a continuous cycle manner. The $^3$He injection line is also above the pot, and consists of a small diameter tube that empties onto a small solid metal cone for condensation purposes. The liquid $^3$He drips off of the tip of the cone and falls into the pot, and the pot cools through evaporative cooling.

The other pumping line indicated in the figure is the evacuation line for the can enclosing the sample. This line is normally roughed out with a mechanical pump and then pumped out with a diffusion pump. Once the vacuum can is near 4.2K, the diffusion pump is not needed, since there is also activated charcoal inside the can, and this acts as a very effective cryopump.

The electrical lines for the probe travel through one of the pumping lines to the top of the probe, where they travel through a 20-pin glass electrical feedthrough. The 20-pin connector is then translated to a 25-pin male RS-232 connector. The pinout for this connector is given in Figure 6.

Figure 7 gives the diagram of the $^3$He / $^4$He pumping system used in the experiment. This system was primarily used in a $^3$He refrigerator mode.

b. Description of Data Collection System (Hardware)

A computer was used in this experiment as the primary data collector, eliminating the need for a dedicated pen and ink x-y plotter. Its main purpose was to automate the electronic aspects of the specific heat probe data collection system, and store the accumulated data. It also facilitated the rapid off-line analysis of the data, and simplified the Data-Collector-to-Mainframe data transfer loop necessary for
Figure 7
some of the statistical analysis. It allowed the experiment to be rapidly reconfigured without reconfiguring the hardware, and eliminated the need for some specialized hardware, such as an analog Proportional-Integral-Differential (PID) block temperature controller.

Figure 8 gives the layout of the data collection hardware. The main component of this system is an I.B.M. personal computer equipped with a 30 megabyte hard disk drive, two high density floppy disk drives, an 8087 math coprocessor, an Ethernet interface, and a Tecmar Labmaster data collection card. The Tecmar card is of course the item of interest here. It consists of two 12-bit Digital to Analog converters (D/A), 24 digital I/O lines, a sixteen channel multiplexer (configured as two eight channel multiplexers) followed by a programmable gain differential amplifier front-ending a 16-bit analog to digital (A/D) converter. It also includes five programmable 16-bit counter-timers that are used primarily as a time-base providing 100-micro-second precision.

The first D/A converter controls the programmable Harrison power supply that in turn controls the block temperature through a voltage divider connected to the block heater. This heater was made by anti-inductively winding constantan wire around the \(^3\)He pot. The heater is non-inductively wound in order to eliminate the magnetic field that would be created by the heater current. The voltage range and resolution of the first D/A converter is set at +/- 10 volts, with a resolution of approximately 0.005 volts. The second D/A converter controls the strain gauge heater, also through a voltage divider. The voltage range and resolution of the second D/A converter has been set to +/- 2.5 volts, with a resolution of
Figure 8
approximately 0.001 volts, allowing somewhat finer control of the sample heater.

Sixteen of the twenty-four digital I/O lines are dedicated to reading and controlling the S.H.E. conductance bridge. This device, referred to hereafter as the S.H.E., measures the conductance of the block thermometers. The information presented on the digital I/O lines coming out of the S.H.E. represents the scale, time constant, excitation level, various hardware flags (e.g. 'conversion-complete'), and digital conductance value calculated by the bridge. The other eight I/O lines are unused in this experiment. The I/O port and pin assignment map is given in Figure 9.

The timer-counter chip on the Tecmar board is configured internally, via software control, to count seconds in 100 micro-second increments. No external connections to the AMD-9513 timer chip are used in this experiment, although the counter values are available through the I/O ports of the Tecmar board.

The A/D converter was set up as eight differential input, programmable gain channels, with the following assignment map:

#1  Analog output of the LIA
#2  Strain-gauge heater voltage
#3  50mv magnetic field shunt
#4  Analog out of the S.H.E. conductance bridge
#6  Analog out from fixed-point calibration LIA
The amplifiers were set up as differential input in order to provide maximum common mode rejection of 60-cycle noise present in the lab, as well as rejection of any D.C. offsets present in the instrumentation due to different system grounds.

The S.H.E. self-balancing conductance bridge is used primarily to measure the conductance of the germanium resistors residing on the block. It performs this through a four-wire A.C. measurement of the conductance by keeping a constant A.C. (27hz.) excitation of approximately 100 micro-volts across a Wheatstone bridge containing the resistor. It then balances the bridge automatically to minimize power dissipation in the resistor being tested. Under some situations it was also used to measure the conductance of a 1k-ohm or 5k-ohm chip resistor that was in series with the germanium resistor (using a six-wire setup), providing a secondary block thermometer. The conductance bridge itself has been modified away from the factory model by providing an analog output of the conductance that was fed to the Tecmar analog to digital converter board. This proved necessary because of the poor accuracy and poor reliability of the digital panel meter contained in the bridge that provided the digital conductance value (worse than 2%). The analog output of the bridge was used for the calculation of the conductance, while a majority of the digital signals were discarded. Some of the digital signals were still important, however, since they provided information on the switch and scale settings of the bridge. These signals were brought out of the bridge via an interface box that interfaced the S.H.E. to the digital I/O ports of the Tecmar board. The schematic of this box is given in Figure 10.
OUT from PC to input of LS244

PORT B

IN to PC and and out from LS244

PORT A

36 pin Centronics connector on SHE

N/C • 38 18 • N/C

0.01 mv • 35 • 0.2 mmho
0.03 mv • 1 sec
20 mmho • 15 • 2 mmho
10 sec • 0.3 sec
200 mmho • 0.1 mv
3 sec • SW comm
10 • 0.3 mv
20 • 4
10 • 2
 Data Ready • 1
Hold • 20 • 10,000
 Data comm • 19 • Overflow

Box shows 34 pin connector on circuit board

Figure 9
The Keithley 179A Digital Volt-Ohm-Current Meter (D.V.M.) was used primarily to give a visual indication of the reading of the 100k-ohm block resistor. This was used to check the temperature of the block during cool-down from liquid nitrogen temperatures to liquid helium temperatures.

The E.G.&G. P.A.R. model 5209 Digital Lock-In Amplifier (D.L.A.) was used to measure the resistance of the sample thermometer. This was accomplished by sending the reference output of the DLA through a voltage divider, and sensing (via another four-wire measurement) the voltage across the chip thermometer. The DLA proved to be very stable with respect to temperature and time drift, so readjustment of phase and reference amplitude during the course of a run proved to be essentially unnecessary. The normal operating levels were less than 0.001 volts across a 2k-ohm resistor, indicating a measurement power of less than $10^{-9}$ watts. The analog output of the DLA was then sensed by the A/D section of the Tecmar board.

As an aside, the original implementation of the data collection system used an analog lock-in amplifier. This amplifier was prone to temperature drift, time drift, phase drift, and output amplifier non-linearities. Also, the offset controls were prone to be noisy due to the use of ten-turn potentiometers. This made it especially difficult to calibrate the sample thermometers, since one was forced to use the same instrument settings on all measurements. With the replacement of this lock-in by the D.L.A., the accuracy and reproducability of the specific heat measurements was increased by a factor of five.
Figure 10
The Harrison power supply received its programming voltage from the #1 D/A converter. The output of the power supply then went through a voltage divider (a ten-turn 2k-ohm potentiometer) and the output of the voltage divider drove the block heater directly.

c. Description of Data Collection System (Software)

Figure 11 presents a flow-chart of the data collection software. The first block, Initialization, performs all of the variable initializations. This is accomplished by a menu-driven front end. The bolometer and block thermometer constants are then assigned, the statistical sums are initialized, the temperature routines are initialized, the assembly routines for the SHE and the AMD-9513 are run (the initialization routines), and the display screen is drawn.

At this point, the heat-leak calibration procedure is started. First, the bolometer heater is set at the first power step (usually zero voltage). Then the main "data point is collected" routine is called. This will be expanded on later. The Step termination condition is examined; i.e., is the temperature vs. time slope equal to zero, within the error conditions? If not, another "data point is collected." If the slope is "zero," then the statistical sums, averages, standard deviations, etc., are calculated and stored, since a steady-state equilibrium has been established. If this is not the last step, then the program returns to the step block. If it is the last step, the program then calculates the linear power fit vs. temperature squared, and stores these constants.
Next comes the re-initialization of the statistical sums, and the termination of sample heating. Then another "data point is collected." The A/D values are recorded, and again a check is performed to see if the relaxation curve has flattened out to "zero" slope. If so, then the program stops recording data, and closes the data file, else it returns back to a "data point is collected."

The above is of course a gross outline. The actual program is somewhat more complicated. Specifically, the "data point is collected" routine is actually a major routine that is responsible for calling the A/D routine, the timer routine, the temperature calculation routines, the block temperature control routine, the display routine, the termination condition routine, and several other important routines, including the time-slicing of the experiment.

The temperature calculation routines are of some interest here since these had to be optimized for speed. The 1k-ohm chip thermometers normally satisfied the following type of equation:

$$\log_e(R) = \sum_{i=0}^{M} A_i T^{(i+1)}$$

where $R$ is the resistance of the chip resistor in ohms, and $T$ is the temperature in Kelvin degrees. This is a slight modification of the functional form used in the Q. Li et al. paper, although this form was used normally in calculations for the 100k-ohm resistors. The reason for using this form for the temperature calculation was to allow a broader temperature range with one bolometer temperature sensor calibration. Figure 12 gives a typical calibration data set for a 1k-ohm chip resistor, and figure 13 gives the deviation of the temperature for a given temperature fit. The temperatures for the above data were derived from the calibrated germanium
For the 10k-ohm and the 100k-ohm chip thermometers, the above equation was modified slightly to allow fewer overall terms via optimization of the exponent used on the temperature variable; i.e., the fitting equations were

$$\log_e(R) = \sum_{i=0}^{k} A_i T^{-\alpha}.$$  

The value of the 'X' exponent was determined via iteration in order to develop the most linear form for the above equation, i.e. limiting the sum over i to just two terms. This allowed for a very fast determination of the sample temperature. This was necessary for the higher temperature (> 5 K) measurements of the specific heat because of the very short decay times involved.

There were two germanium temperature sensors used in these experiments, for two different temperature ranges. The germanium thermometers' conductance values satisfied two different fitting equations. The high temperature germanium thermometer, used in the 1.2K to 25K range, was fit to the following functional form:

$$\log_{10}(R) = \sum_{i=1}^{10} B_i T^i.$$ 

This form was used in the 1.2K to 25K range, although there were two sets of coefficients: one set for the 1.2K to 5K range, and one set for the 5K to 25K range. The low temperature germanium thermometer, used in the 0.5K to 5K range, was fit to the following functional form:

$$\log_e(R) = \sum_{i=0}^{12} C_i (\log_e(T))^i.$$ 

These fitting equations were strictly empirical fits that normally had ten or more terms.
Figure 13
When more than three terms are involved, the expression for the temperature is difficult, or impossible, to arrive at in a closed form; therefore, iterative methods of root-finding have to be implemented. Specifically, the standard approach was to use a Newton-Raphson Method of iteration on a polynomial written according to Horner's rule. This yielded an improvement over a previously implemented numerical bisection method on a standard 14th order polynomial of approximately 500:1, when the polynomials were optimized for speed. A major factor contributing to this speed increase is that Newton’s method has second order convergence whereas bi-section is first order. The disadvantage is of course that the interpretation of the error bounds is more difficult for Newton’s Method, so that it is necessary to install an 'IF' statement to determine the fastest termination condition. The method of bi-section, by contrast, gives a guaranteed error condition after 'N' iterations. In most of the temperature routines used in the data collection programs, the termination condition was 10^3 Kelvin: for the high temperature germanium thermometer used in the 1.2 to 25 Kelvin, bisection would require 18 iterations for convergence, while Newton’s method averages 5.6 iterations for convergence in normal operation (with an initial temperature guess of 5 K), which is a clear advantage. However, there is a disadvantage in the use of Newton's Method in that it suffers from instabilities in the presence of local maxima and minima. As long as the equations being iterated are monotonic, this instability is relatively unimportant.

A modification of this iteration scheme was pursued in order to increase the loop speed even further. Normal Newton’s method iteration fits the form of the
Instead of this scheme, a new iteration method was derived by defining a new function \( u_n \) represented by
\[
 u_n = f(x_n)f''(x_n)/f'(x_n)^2 ,
\]
and then a new iterative procedure given by the following scheme was used (see appendix A for derivation):
\[
 x_{n+1} = x_n - f(x_n)/f'(x_n) \left[ 1 + (1/2)u_n + (1/2)u_n^2 + \ldots \right] .
\]
This iteration formula turned out to be faster, and has the same requirement that Newton's method has in that it requires only that \( f' \) be non-zero.

With the installation of the above iteration formula, the average number of iterations for convergence with an initial guess of 5K was 3.7, and the overall computational speed increase was approximately 15%. This scheme was more prone to instabilities, however, and could not be effectively used with the low temperature germanium thermometer due to slight fluctuations in \( f''(x) \) in the temperature fitting routine. This indicates that the low temperature germanium thermometer should have a new fitting function derived for it, eliminating the oscillations in \( f''(x) \).

This relentless pursuit of computational speed in the block temperature calculation was not merely an exercise in computational stream-lining. There is a very real need to calculate the block temperature quickly in order to facilitate the block temperature control algorithm discussed in Appendix A. An increase in the speed of the temperature calculation allows an increase in the system sampling rate, allowing a finer temperature mesh for the specific heat calculations as well as a
finer time resolution. It also allows the PID control algorithm to more closely approximate an analog controller, thus allowing lower (temperature) noise oscillation levels inherent in the block temperature controller.

Most of the data collection programs were written in BASIC in order to take advantage of its interactive graphics and I/O capabilities, although critical subroutines were written in 8086 ASSEMBLY language, or FORTRAN '77 and translated to BASIC (they still operate at ASSEMBLY language speeds). The BASIC's used were Microsoft's 87BASIC and 87INLINE BASIC (both are, of course, compiled languages in which 87BASIC has approximately 50% of the speed of 87INLINE). More recent modifications to the data collection routines use Microsoft's QuickBasic 4.5, which overall has approximately 90% of the speed of 87INLINE BASIC, but with a much improved user interface.

d. Discussion of Thermometer Calibration Procedures

Before a data-run, the thermometers in the system must be calibrated. This is normally accomplished in a separate data-run, by anchoring the sample thermometer to the block with varnish in order to achieve good thermal contact. The germanium block resistor is used as the calibration standard for the various Bi-Ru chip sample thermometers. The block temperature is swept over the range of interest (for 1k-ohm resistors, this is normally 0.5K to 5K) at a slow pace in order to minimize heat gradients in the block. The germanium thermometer temperature and the sample thermometer resistance are continuously collected, and when the
standard deviation of 'N' points satisfies our convergence criterion, the average and standard deviation of those 'N' points is recorded as a single calibration point. Typically, a calibration file contains several hundred calibration points covering the range of interest, and takes up to six hours to collect.

The germanium temperature and the sample thermometer resistance are then plotted against each other and a functional form is chosen to fit the range of interest. Least-Squares or Singular-Value-Decomposition fitting to this functional form is then performed, and the parameters for the particular resistor are then entered into the data collection program and stored in a data file.
CHAPTER III

SPECIFIC HEAT AND MAGNETIC ORDER IN Gd$_2$Ba$_2$Cu$_3$O$_7$

1. Discussion of the Experimental Results for Zero Applied Field, Zero Sr.

Figure 14 shows the specific heat divided by temperature versus temperature measurements on a 10mg Gd$_2$Ba$_2$Cu$_3$O$_7$ sample in zero magnetic field. Points of interest on this curve are:

1) the A.F.M. ordering peak at 2.24 K.

2) the flat shelf region below the ordering peak, where $C_p$ is proportional to $T$.

3) the linear portion of the $C_p/T$ curve that is just becoming visible at low temperature.$^{52}$

4) the low value of the specific heat on the high temperature side of the ordering peak.

Note first that the low value of the specific heat at 3 K, coupled with the high temperature data of Simizu et al.$^{53}$, places an upper limit on the lattice and electronic contribution to the specific heat that is much smaller than the flat region of the specific heat curve below $T_N = 2.24$ K. That is, the flat region represents at least an order of magnitude increase to the specific heat due to magnetic interactions, rather than conduction electrons. Also, the magnetic contribution is much larger than the $T^3$ contribution due to the lattice. Therefore, in the low temperature region, the electron and phonon contributions to the specific heat are
Figure 14
negligible compared to the magnetic contributions. This can be seen by approximating the specific heat contributions due to the phonons and conduction electrons by

\[ c_{\text{sp}} = \gamma T + \alpha T^3 , \]

yielding

\[ \frac{c_{\text{sp}}}{T} = \gamma + \alpha T^2 . \]

Therefore, this term, on the order of 0.425 J/Mol-K^2 at 25 K, gets much smaller as T approaches 0 K. Looking at Figure 14, one sees that the flat region, or shoulder, is on the order of 6.5 J/Mol-K^2 near 1 K. Therefore, the magnetic contributions are at least two orders of magnitude greater than the electron and phonon contributions.

2. Discussion of Spin Wave Model.

The Gd compound is unique among the rare-earth (RE) high T_c superconductors in that the magnetic ion is in an S state. Therefore, the local anisotropy of the crystalline field has little effect on the ground state of the f electrons, since this state is spherically symmetric. A comparison of the specific heat data below the ordering temperature for the RE compounds shows that Gd is quite different because there is a large temperature region not found in the other RE's over which the specific heat is almost linear in temperature (the flat region in Figure 14). In the experiments that have been reported, Gd is also unique among the RE's in that when one changes from GdBa_2Cu_3O_y to GdBa_2Cu_3O_x, and the sample changes from superconducting to insulating, the shape of the specific heat
curve and the ordering temperature are unchanged.\textsuperscript{28}

While dipolar coupling may be responsible for the coupling in the a-b plane, it is difficult to see how it could be responsible for the overall 3D AFM structure, since it would be very weak in the c-axis direction. This point of view also was taken by Liu\textsuperscript{26} who developed a model of RKKY coupling in the c direction involving d bands and f-d exchange on the RE ion.

In order to analyze specific heat data, a model developed by Dana Browne was used. Assume the following spin Hamiltonian:

$$H = \frac{1}{2} \sum_{ij} J(R_i - R_j) S_i \cdot S_j$$

where $S_i$ is a spin-7/2 operator at site $i$. In his analysis he chose the coupling $J(R_i - R_j)$ as follows:

(1) $J_1 > 0$ for two nearest neighbor spins along the c-axis;

(2) $J_2 > 0$ for two nearest neighbor spins along the a-or b-axis (ignore the slight anisotropy caused by the orthorhombic structure);

(3) $J_3$ (of either sign) for the second neighbor spins in the a-b plane, that is, along the face diagonals.

The first two couplings are required to achieve 3D AFM order, since this lowers the total energy. The last term was included because this coupling might be as large as $J_1$, since 2nd neighbors in the a-b plane are actually closer than the nearest neighbors along the c-axis.

Because Gd\textsuperscript{3+} has a large (s=7/2) pure spin moment that is affected by the crystal field only in second order, the low temperature magnetic behavior should be described accurately by spin-wave theory. At low temperatures the spin-wave
spectrum is given by

\[ E = S\sqrt{[\bar{J}_A(0) + \bar{J}_P(0) - \bar{J}_F(q)]^2 - [\bar{J}_A(q)]^2} \]

where

\[ \bar{J}_A(q) = 2J_1 \cos(q_x a) + 2J_2 \cos(q_y a) \]

and

\[ \bar{J}_F(q) = 4J_2 \cos(q_x a) \cos(q_y a) \]

To analyze the behavior at higher temperatures, Browne used a renormalized spin wave theory, similar to that of Bloch and Nakamura, and then performed a mean field analysis; i.e., he replaced the interacting magnons with a set of effective non-interacting magnons in such a way that the free energy was minimized. This produces the following Hamiltonian:

\[ \mathbf{H}_{\text{MF}} = \sum \varepsilon_k (a_k^\dagger a_k + b_k^\dagger b_k) + \Delta_k (a_k^\dagger b_k^\dagger + a_k b_k) \]

Here \( \varepsilon_k \) and \( \Delta_k \) are determined self consistently by the following set of equations:

\[ \varepsilon_k = S[\bar{J}_A(0) + \bar{J}_P(0) - \bar{J}_F(k)] 
- (1/N)\sum_{k'} [\bar{J}_A(k) - J_F(k') - J_F(k - k')] n_{k'} + \bar{J}_A(k)n_k \]

\[ \Delta_k = S\bar{J}_A(k) - (1/N)\sum_{k'} [\bar{J}_A(k - k')\phi_{k'} + \bar{J}_A(k)n_k] \]

where

\[ n_k = (1/2)(\langle \varepsilon_k / E_k \rangle \coth(E_k/2T) - 1) \]

\[ \phi_k = - (\Delta_k / 2E_k) \coth(E_k/2T) \]

\[ E_k = [\varepsilon_k^2 - \Delta_k^2]^{1/2} \]

In the analysis, the above equations are iterated, using the special point scheme of Chadi and Cohen to do the sums over a face-centered tetragonal
magnetic Brillouin zone. A spacing of $\pi/16a$ is used in the $a$-$b$ plane and $\pi/16c$ is used along the $c$ axis. After iterating the above to convergence, the entropy is calculated from the expression

$$S = \sum f_k \ln(1+f_k) - f_k \ln f_k,$$

where $f_k$ is the Bose distribution function,

$$f_k = \frac{1}{e^{\frac{\mu_k}{kT}} - 1}.$$

The entropy is then differentiated numerically to yield the specific heat.

Some of the calculated curves are shown in Figure 15. It therefore seems that the spin coupling is much stronger along the $c$ axis than in the $a$-$b$ plane. The best fit parameters from Browne's model are $J_x = 70 \pm 20$ K, $J_{xy} = 4 \pm 0.4$ K, $J_2 = 0.2 \pm 0.2$ K. With a lower strength $c$ axis coupling, it was not possible to reproduce the flat region in the $C_p/T$ curve of Figure 14.

3. Strontium Substitution.

Two additional samples were prepared for this experiment. They were Sr substitutions for Ba in the Gd$_1$Ba$_2$Cu$_3$O$_7$ compound; i.e., the formula was for the compound Gd$_1$Ba$_{2-x}$Sr$_x$Cu$_3$O$_7$, where $x$ took on the values 0 (pure Ba), 0.25, and 0.5. It turned out that higher values of $x$ yielded a non-superconducting sample. For these values of $x$, the superconducting transition temperatures were measured via A.C. susceptibility to be 90 K, 80 K, and 50 K, respectively.

The purpose of this substitution was to study the role that the barium ions play in the mediation of the interaction of the magnetic moments in the $c$-direction. When looking at the results of the previous section, one sees that the strongest
Figure 15
coupling constant is $J$, yet the Gd ions are three times farther apart than are the Gd ions in the x-y plane. This indicates that some type of super-exchange must be occurring through some intermediary ions. One possibility is an exchange occurring through the barium ions, although this path is relatively complex. Because there are no intermediary ions between the Gd ions in the a-b plane, these ions are most likely interacting through the relatively weak dipole-dipole interaction in this plane.

The specific heats divided by temperature of the three samples were measured near the $x = 0$ Neel temperature, and the results are shown in Figures 4, 16, and 17. All of the measurements were performed in zero magnetic field. In general, the peak height is suppressed, but the position of the peak is relatively unchanged for $x > 0$. In addition, the temperature region below $T_N$ where a linear $C_p$ is observed to be smaller in the $x = 0.25$ sample and practically eliminated in the $x = 0.5$ sample, with $C_p$ decreasing more rapidly in both of the Sr doped samples than in the undoped sample.

The entropy change was calculated from the specific heat curves for the three samples (same method as in Chapter IV). The contribution to the entropy below 0.6 K for the pure barium sample was calculated from the data of Willis et al. Since $C_p/T$ is much lower for the Sr doped samples, the low temperature portion of the curve was extrapolated smoothly to $T = 0$ for these entropy calculations.

For the ordering of spins in zero magnetic field, one would expect an entropy change of

$$\Delta S = R \ln(2S+1)$$
Figure 16
This comes about from using a definition for the entropy of

\[ \Delta S = - R \sum_M P_M \ln P_M, \]

where \( P_M \) represents the probability that the spin will be in state \( M \). For a free spin, this probability is uniformly distributed over the \( 2S + 1 \) states (since they are degenerate); therefore, \( P_M = 1/(2S + 1) \). Then,

\[ \Delta S = - R \sum_M P_M \ln P_M = - R[(1/(2S + 1))(2S + 1)] \ln[1/(2S + 1)] \]

\[ = R \ln(2S + 1) = R \ln(2(7/2) + 1) = 17.29 \text{ Joule/Mole-Kelvin} \]

for a spin \( 7/2 \) system. From the experimental data on the various samples, we calculated the following values for the entropy change:

<table>
<thead>
<tr>
<th>Compound</th>
<th>Entropy Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Gd}_2\text{Ba}_2\text{Cu}_3\text{O}_7 )</td>
<td>( \Delta S = 17.2 \pm 1 ) J/Mol-K</td>
</tr>
<tr>
<td>( \text{Gd}<em>{1.75}\text{Sr}</em>{0.25}\text{Cu}_3\text{O}_7 )</td>
<td>( \Delta S = 15.1 \pm 1 )</td>
</tr>
<tr>
<td>( \text{Gd}_{1.5}\text{Sr}_3\text{Cu}_3\text{O}_7 )</td>
<td>( \Delta S = 13.3 \pm 2 )</td>
</tr>
</tbody>
</table>

Note that the entropy loss is approximately in direct proportion to the amount of barium in the sample. For example, the 0.25 Sr substitution has the ratio of the entropy change of the doped sample to the undoped sample of approximately 7/8, and similarly the ratio of the entropy change of the 0.50 Sr sample to the pure barium sample is approximately 3/4. Obviously, the total entropy change in each case must be \( R \ln 8 \), since the 7 unpaired electrons of each Gd ion still exist. One possibility for the inconsistency is that the replacement of the Ba ions by the Sr ions weakens the Gd c-axis coupling significantly, and the freed spins then order at a temperature significantly below the measurement range of the data. Thus,
Figure 17
extrapolating the specific heat curves smoothly to 0 is not sufficient, since the remaining entropy would be in error due to an unknown shape for a low temperature peak. It is possible that the slight upturn in the low temperature portion of the 0.50 strontium specific heat curve is part of this low temperature peak; however, the uncertainty is large enough at this end of the specific heat curve (approximately 10%) that the confidence in this interpretation is not high.

The net result of this substitution is that the substitution of Sr for Ba results in a proportional reduction of the entropy change. This indicates that the Gd moments are not all ordering near the Neel temperature = 2.24 K, and are in fact ordering outside of the range of measurement. Thus, the strength of the c-axis coupling has been reduced, freeing some of the Gd moments, indicating that the Ba ions are indeed the probable mediators for the super-exchange, or are an intermediary ion for the exchange. The Neel temperature is also unaffected; however, this can be explained by the presence of local three dimensional order in the crystallites for the remaining Gd ions that are not freed by the strontium substitution.
CHAPTER IV
SHIFT IN ANTIFERROMAGNETIC ORDERING TEMPERATURE
WITH APPLIED FIELD

1. Mean (Molecular) Field Theory

In order to understand the shift in the Neel temperature $T_N$ observed in this experiment, an approach from Molecular Field Theory will be used. This will be used mainly for a qualitative understanding of the phenomena, rather than for a truly quantitative picture.

Consider the Heisenberg Hamiltonian for a magnetic crystal in an external field given by the following:

$$ H = \frac{1}{2} \sum_{R, R'} J(R - R') S(R) \cdot S(R') - H \sum_{R} S_z(R) . $$

Here $J(R - R')$ represents the interaction integral between the $R$ and $R'$ Bravais lattice sites. If one focuses on a particular lattice site $R$, the Hamiltonian for the field at site $R$ can be represented as

$$ \Delta H = \left[ \sum_{R'} J(R - R') S(R') \right] \cdot S(R) - H \cdot S(R) . $$

or define an effective "field" acting on the $R$ site as

$$ \Delta H = H_{\text{eff}} \cdot S(R) . $$

This effective field is a result of the sums of the local magnetizations at the $R'$ sites and of the externally applied field.

An antiferromagnetically ordered lattice (for the assumed form of the Gd crystal as a simple tetragonal crystal) can be considered as two interpenetrating
ferromagnetic lattices, A and B (see 2-D XY plane representation in Figure 18).

Thus, the effective field at an A site $R$ can be given by

$$H_A = \sum_{R' \in A} J(R - R')S(R') + \sum_{R' \in \text{B}} J(R - R')S(R') - H.$$ 

Similarly, for the B sites $R$

$$H_B = \sum_{R' \in A} J(R - R')S(R') + \sum_{R' \in \text{B}} J(R - R')S(R') - H.$$ 

Approximate the total spin angular momentum of an individual ion by replacing it with its thermal equilibrium value (the Mean-Field approximation), which is a function of the total magnetization of the particular lattice (the magnitude of which is the same for all sites). Approximating the a and b axes to be the same (tetragonal approximation) leads one to the result that the four nearest neighbors of a site $R$ in the a-b plane (residing on the other lattice) all have the same coupling constant $J(R - R')$. Define this constant to be $J_\alpha$. Similarly, the four second-nearest neighbors in the a-b plane (residing on the same lattice as the $R$ site) will have the same value for the coupling constant $J(R - R')$. Define this constant to be $J_2$. In the c-direction, the two nearest neighbors (residing on the other lattice) will have the same value; therefore, define this constant to be $J_z$ (see figure 19). A restriction of the sums over the first and second nearest neighbors results in the following expressions for the effective fields

$$H_A = -H + (2J_\alpha + 4J_\alpha)M_B + 2J_2M_A,$$
and

\[ H_B = -H + (2J_x + 4J_y)M_A + 2J_zM_B. \]

The Mean Field approximation is now applied; i.e., the Maxwell-Boltzmann distribution of the spins gives (for a spin \( j \) system):

\[ M = \frac{N\sum_{j} g\mu_B e^{M\mu_B H/kT}}{\sum_{j} e^{M\mu_B H/kT}} \]

and if one lets \( x = g\mu_B H/kT \), this gives

\[ M = N\mu_B \frac{d}{dx} \ln \left[ \sum_{j} e^{Mx_j} \right]. \]
The above sum is a geometric series that can be readily summed to give the following result for the average value of the spin:

\[ M = N \mu_a \int \frac{d}{dx} \ln \left\{ \frac{\sinh[(2j+1)/2]x}{\sinh(x/2)} \right\} \]

\[ = N \mu_a \left\{ [((2j+1)/(2j)) \coth[(2j+1)/(2j)]y - (1/2j) \coth[y/(2j)]] \right\} \]

\[ = N \mu_a B_j(y) \]

where

\[ y = jg\mu_B H/kT \]

\( B_j(y) \) is usually called the Brillouin function. A graph of the Brillouin function for a spin \( 7/2 \) system, \( B_{7/2}(x) \), is given in Figure 20. With this form for the spins, the equations for the magnetizations can now approximately be written as

\[ M_A = \frac{1}{2} N \mu_a B_j [jg\mu_B (H - (2J_z + 4J_{xy})M_B - 2J_2M_A)/k_B T] \]

and

\[ M_B = \frac{1}{2} N \mu_a B_j [jg\mu_B (H - (2J_z + 4J_{xy})M_A - 2J_2M_B)/k_B T] \]

where the substitution

\[ <S(R)> = (V/N)M/g\mu_B \]

has been applied for the relation between the thermal equilibrium value of the spins and the magnetization. The \( (1/2)N \) factor means half of the total lattice is aligned in one direction, and the other half is aligned antiparallel to this lattice.

In order to find the AFM ordering temperature, \( T_N(H) \), the above equations must be searched for solutions such that \( M_A = -M_B \), i.e. total magnetization \( M = 0 \). Define two new parameters \( P \) and \( M \) such that

\[ P = (M_A - M_B)/2 \]

and
Then the ordering temperature occurs when $M = 0$ and $P \neq 0$. The full expressions for $M$ and $P$ are given by

$$M = \frac{1}{2} N \beta B_{72} [2 \beta (H - (K + K')M + (K - K')P)/k_B T]$$

$$+ \frac{1}{2} N \beta B_{72} [2 \beta (H - (K + K')M - (K - K')P)/k_B T] ,$$

and

$$P = \frac{1}{2} N \beta B_{72} [2 \beta (H - (K + K')M + (K - K')P)/k_B T]$$

$$- \frac{1}{2} N \beta B_{72} [2 \beta (H - (K + K')M - (K - K')P)/k_B T] ,$$

where

$$K = 2J_4 + 4J_{xy} , \quad K' = 2J_2$$

**Figure 20**

$$M = (M_a + M_b)/2 .$$
and

\[ \beta = (J/2)g\mu_B . \]

A closed form solution is not possible, so further approximations will have to be made. One possibility is to use an expansion for the Brillouin function for small values of x. That is, can one use the following expansion?\( B_j(x) = [(j + 1)/3j] \{(x - x^3)((2j^2 + 2j + 1)/(30j^3))\} \).

This expansion is allowed, provided the denominator in the Brillouin function expressions for M and N is bigger than the numerator. For the determination of the Neel temperature \( T_N(H) \), the value of M should be 0. Therefore, the numerator of the Brillouin function is \([2\beta \{H \pm (K - K')P\}]\). It is this value which should be smaller than \( k_B T \). At 1 Tesla, \( \mu_B H/k_B = 0.67\) Kelvin, which is significantly smaller than the observed transition at 2K. Note that since P is the difference of two Brillouin functions, the maximum value of which is 1, one can assume that the entire numerators of the Brillouin functions are much smaller than the denominators.

Since the main interest is in small magnetic field behavior, the above approximation should be fairly good. Therefore,

\[ P = (1/2)N\beta[(J + 1)/3J]\{2\beta[H + (K - K')P]/k_B T \]

\[ - \alpha[2\beta(H + (K - K')P)/k_B T]^3 \]

\[ - [2\beta(H - (K - K')P)/k_B T] \]

\[ + \alpha[2\beta(H + (K - K')P)/k_B T]^3 \} , \]

where \( \alpha = (2j^2 + 2j + 1)/30j^3 \). If one simplifies the above, one gets

\[ P = P\beta N[(J + 1)/3J][2\beta(K - K')/k_B T]\{1 - 3\alpha(2\beta H/k_B T)^2 - O(P^3)\} . \]

If one keeps terms only to order \( O(P^3) \), the above expression for P reduces to
\[ P = P_{\text{PNK}} + \frac{l}{3J} \left[ 2\beta (K - K')/k_B T \right] \{ 1 - 3\alpha (2\beta H/k_B T)^2 \} \]

or, dividing out the quantity \( P \),

\[ 1 = \beta N[(J + 1)/3J][2\beta (K - K')/k_B T][1 - 3\alpha (2\beta H/k_B T)^2] \]

At zero applied field, this gives a value for the Neel temperature of

\[ k_B T_N(0) = 2\beta^2 (K - K')(J + 1)/3J N \]

Note that using the previously obtained values for \( K \) and \( K' \), and for \( J = 7/2 \), this gives a zero-field ordering temperature of

\[ T_n(0) = \beta (K - K')(9/21)/k_B = 67 \text{ K} \]

Needless to say, this value is not accurate (measured \( T_n(0) = 2.24 \text{ K} \)), which indicates that mean-field theory is not an accurate representation of the true picture.

If one uses this in the expression for a non-zero applied field, one gets

\[ k_B T_N(H)/k_B T_N(0) = 1 - 3\alpha (g\mu_B H/k_B T_N(H))^2 \]

For an order of magnitude calculation, change the above cubic equation (in terms of \( T_N(H) \)) into a linear equation by approximating the denominator \((k_B T_N(H))^2\) as \( (k_B T_N(0))^2 \); i.e., assume

\[ [T_N(H) - T_N(0)]/T_N(0) \ll 1 \]

Then the expression reduces to

\[ T_N(H)/T_N(0) = 1 - 3\alpha (g\mu_B H)^2/(K - K')(J + 1)/3J] \]

If one substitutes the value 7/2 for \( J \), and substitutes for the value of \( \alpha \), one gets

\[ T_N(H)/T_N(0) = 1 - (9/10)(2J^2 + 2J + 1)/(J + 1)^2 (g\mu_B H)^2/(K - K')^2 \]

\[ = 1 - (13/9)(g\mu_B H)^2/(K - K')^2 \]

\[ = 1 - (13/9)(g\mu_B H)^2/(2I_x + 4J_{xy} - 2J_2)^2 \]

\[ = 1 - (g\mu_B H)^2/[(3/13)^2(2I_x + 4J_{xy} - 2J_2)^2] \].
2. Experimental Results for Gd$_1$Ba$_2$Cu$_3$O$_7$

The specific heat of a 10mg Gd$_1$Ba$_2$Cu$_3$O$_7$ sample was measured from 0.8K to 3.0K in magnetic fields from 0 to 5 Tesla in a super-conducting magnet. A composite figure showing the specific heats for the fields ranging from 0 to 2.0 Tesla, in 0.5T increments, is shown as Figure 21. Note that the peak position (the Neel Temperature $T_N(H)$) is suppressed with increasing magnetic field. It turned out that the ordering peak was unobservable for field values greater than 2.0T, so these curves are not shown. Also note that the peaks are broadened, possibly due to the random orientation of the crystallites of the sintered powder sample.

The temperatures of the peak positions, $T_N(H)$, were measured and recorded. These temperatures were then plotted against $H^2$, and the results of that plot are given in Figure 22. Van der Meulen et al. measured GdBa$_2$Cu$_3$O$_7$ in fields to 5 T; however, their results could only show the 0 T and 1.0 T ordering peaks. Their values for these peak positions were 2.23 K and 2.05 K respectively. No discussion of system accuracy was given in their paper. Our results for these two peaks were $2.235 \pm 0.03$ K and $2.031 \pm 0.03$ K, very much in agreement with their values. Note that the functional dependence is very close to linear; therefore, a least-squares line was fit to the Neel temperatures. The least-squares results were

$$ T_N(H) = b + aH^2 $$

$$ a = -0.228 \pm 0.004 \text{ K/T}^2 $$

$$ b = 2.241 \pm 0.008 \text{ K} = T_N(0) $$

or, if one divides a by b,
Figure 21
\[ T_n(H)/T_n(0) = 1 - (0.102 \text{T}^2)H^2 \]
\[ = 1 - (g\mu_B H)^2/[g\mu_B (3.14 \text{T})]^2 , \]
where the unit 'T' represents Tesla. When the denominator in the above expression is changed to temperature units, and the actual effective magnetic moment for Gd is used, the equation reduces to
\[ T_n(H)/T_n(0) = 1 - (\mu_{\text{eff}} H)^2/[k_B (16.5 \text{K})]^2 . \]
Note that the denominator of the field dependent term is quite a bit smaller than the 70 K coupling constant value from mean field theory.

\[ \text{Figure 22} \]
The final result is that the coupling constants should be on the order of 2 K, instead of 70 K, if the mean-field theory were an accurate model. Therefore, the mean-field calculation represents a poor model for the calculation of the Neel temperature, since it cannot reproduce order-of-magnitude results, only the sign of the coefficient. However, with re-scaling of the mean-field values with the correct form for the $T_N(0)$ value, one obtains the following expression for the field dependence

$$T_N(H)/T_N(0) = 1 - 3\alpha(\mu_\text{eff}H)^2/T_N(0)^2$$

$$T_N(H)/T_N(0) = 1 - (1/10)((2I^2 + 2J +1)/I^2)(\mu_\text{eff}H)^2/(k_b2.24 \text{ K})^2$$

$$= 1 - (13/49)(\mu_\text{eff}H)^2/(k_b2.24 \text{ K})^2$$

At least this expression is order-of-magnitude correct. However, mean field theory does indicate that the negative field-squared dependence is there, and this is an indicator of antiferromagnetism.

<table>
<thead>
<tr>
<th>Peak</th>
<th>Field</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.235 K</td>
<td>0.0 T</td>
<td>17.9 J/Mol-K</td>
</tr>
<tr>
<td>2.175</td>
<td>0.5</td>
<td>17.9</td>
</tr>
<tr>
<td>2.031</td>
<td>1.0</td>
<td>13.6</td>
</tr>
<tr>
<td>1.730</td>
<td>1.5</td>
<td>16.0</td>
</tr>
<tr>
<td>1.325</td>
<td>2.0</td>
<td>15.1</td>
</tr>
<tr>
<td>----</td>
<td>2.5</td>
<td>11.2</td>
</tr>
<tr>
<td>----</td>
<td>3.0</td>
<td>11.9</td>
</tr>
</tbody>
</table>

Figure 23
The change in entropy was also computed for these field dependent curves, with the results given in figure 23. These values were calculated by

$$\Delta S = \int_{0.8}^{3.0} (C_p/T) dT + \Delta S_{\text{tail}}$$

where $\Delta S_{\text{tail}}$ represents the entropy calculated from the Willis et al.\textsuperscript{68} specific heat at zero-field data below 0.8 K. Note that, in general, the change in entropy change is decreasing from the Rln8 value of the zero-field data. This is most likely indicative that the Gd moments are ordering at a temperature outside of the 0.8 to 3.0 K range of the experiment, so that the use of the tail of the Willis data does not account for all of the entropy change. The tail would have to be modified to account for the downward shift of the ordering peaks. Data need to be taken down to 0.1 K in the 0 to 2 T range in order to get accurate values for the entropy change. This is not currently feasible in the experimental setup.
CONCLUSIONS

The strontium substitution for the barium ions had a proportional affect on the calculated entropy for the 123 samples. This indicates that the Gd moments are not all ordering near the Neel temperature = 2.24 K. It therefore seems that the strong c-axis coupling predicted by Browne's spin-wave model (best fit parameters from Browne's model are $J_z = 70 \pm 20$ K, $J_{xy} = 4 \pm 0.4$ K, $J_2 = 0.2 \pm 0.2$ K), which accurately predicts the low-temperature shoulder in the $C_p/T$ curve, is indeed mediated by the barium ions. This is further substantiated by the loss of the low-temperature shoulder in the $C_p/T$ curve as the strontium substitution goes up. This is reasonable, since replacement of the barium ions weakens the c-axis chains, making it more difficult for spin-waves to achieve long-range order in the c-direction.

The missing entropy must still show up; however, it will most likely show up as a Schottky-like peak at temperatures well below the small sample calorimeter's range. It is possible that the slight upturn in the low temperature portion of the 0.50 strontium specific heat curve is the beginning part of this low temperature peak.

The Neel temperature is not affected by the strontium substitution; however, this can be explained by the presence of local three dimensional order in the crystallites for the remaining Gd ions that are not freed by the strontium substitution.
Although mean-field theory does a poor job in predicting the Neel temperature, since it does not take into account the possibility of any kind of spin fluctuations, nor does it work successfully on almost one-dimensional systems, it does characterize the negative $H^2$ dependence as being a property of this antiferromagnet.

Further work needs to be done in order to fully verify the spin-wave model. One possible avenue of research is energy ($q$) dependent neutron scattering below and above $T_c$ in Gd$_2$Ba$_2$Cu$_3$O$_7$. Below $T_c$, this would fully characterize the spin-wave structure of this system. Above $T_c$, evidence of spin fluctuations or persistence of local magnetic order, indicated by the absorption of energy from the neutrons, would further verify the model. It should be noted here that some evidence for spin fluctuations well above $T_c$ is indicated in the work of Golnik et al.\textsuperscript{69}, which gives credence to the spin-wave model.

Further work at L.S.U. could progress along the lines of lower temperatures (0.05 K) and higher magnetic fields (12 T) for Gd$_2$Ba$_2$Cu$_3$O$_7$ and other rare-earth high temperature superconductors that have local magnetic order, such as Ho, Er, etc.. In addition, the mean field theory, and possibly the spin-wave theory need to be modified to be applied to some of these other magnetic rare-earth systems.
1. Modified Newton's Method Derivation

A modification of the standard Newton’s method iteration scheme was pursued in order to increase the computational speed of the block thermometers. Normal Newton’s method iteration fits the form of the following:

\[ x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \]

This can be derived from an approximation of the standard Taylor series expansion for a function \( f(x) \):

\[ f_{\text{Approx}}(x) = f(x_0) + f'(x_0)(x - x_0) \]

If the above approximation is solved for \( x \), this yields the standard Newton’s method equation. This turns out to be equivalent to approximating the function as a straight line by taking the functional value at \( x_0 \), its derivative, and fitting a line to the above values. The \( x \)-axis intercept is then the next guess.

If one increases the approximation to second order, then the following approximation results:

\[ f_{\text{Approx}}(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)(x - x_0)^2}{2} \]

If the above approximation is solved for \( x \), a new iteration scheme emerges:

\[ x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}[1 - \frac{2f(x_n)}{f''(x_n)}]^{1/2} \]
This form is equivalent to approximating the function with a parabola by taking the functional value at \( x_q \), its derivative, and its second derivative and fitting a parabola to these values. Again, the x-axis intercept is then used as the next guess. Since a parabola of this form will intersect the x-axis in two places, the choice is made to pick the smaller value. This is done because the functional form for the temperature fit is monotonic, and the second derivative does not change sign. This form is not particularly useful, since the calculation of a square root is necessary in the core of the iteration loop. Needless to say, the calculation of any exponential function inside of a (hopefully) tight loop will be slower computationally than a single multiplication or addition operation. Not only does the above iteration scheme utilize an exponential function, but it also requires insurance that both \( f'' \) and \( f' \) must be non-zero, as well as the additional requirement that

\[
[f(x)f''(x)/f'(x)^2] < 0.5 .
\]

in order to avoid an unrealistic square-root of a negative number. The latter possibility exists in the perfectly plausible case that there is no x-axis intercept for some parabolas. In such a case, normal Newton's method iteration should be used. In order to make this iteration scheme more useful, the square root will be approximated via the expansion formula below

\[
(1-2x)^{1/2} = 1 - x - (1/2)x^2 - (1/2)x^3 - (5/8)x^4 - ... 
\]

which is valid for \( x < .5 \). If \( x \) is replaced with the function represented by

\[
u_n = f(x_n)f''(x_n)/f'(x_n)^2,
\]

a new iterative procedure given in the following iteration scheme is developed:

\[
x_{n+1} = x_n - (f(x_n)/f'(x_n))[1+(1/2)u_n+(1/2)u_n^2+...] .
\]
This iteration formula is much more useful, and requires only that \( f' \) be non-zero. Although the above expansion of the square root requires that \( u < 0.5 \), the iteration scheme works (empirically) even when \( u > 0.5 \), obviating the need for determining whether \( u \) is less than 0.5.

The installation of the above iteration formula, utilizing terms up to \( u^2 \) resulted in an average number of iterations for the same high-temperature germanium thermometer of approximately 3.7 (starting from an initial temperature guess of 5 K). This represents an increase in iterative speed of 35%. Not only was this method faster on the average, but it was also faster on every individual temperature calculation. This represented a significant improvement over the old Newton's method scheme, and was fully implemented in the data collection program for the high temperature thermometer. There was additional computational overhead involved in the calculation of \( f''(x) \) and in the computation of the more complex iteration scheme, so that the overall computational speed increase was only 15%.

2. PID Controller Discussion

As stated previously, the need for a computational speed increase was to allow for a faster control loop for the block temperature controller. As a general rule, the faster the sampling rate for a Direct Digital Control loop (DDC), the more closely it approximates the performance of its analog counterpart. This is important since an analog model for the control loop was implemented in order to control the block temperature, and analog tuning techniques were used to fit the controller to the current system.
The control algorithm chosen was that of the discrete equivalent of the Proportional-plus-Integral-plus-Derivative action (PID) controller. This algorithm was chosen since it did not require a carefully constructed model of the system heater and probe, which would have been difficult at best. It also was easier to change the system parameters with a priori knowledge that the system could undergo severe load and set point changes. It allowed for a relatively rapid implementation of block heater control, with the ability to quickly change the parameters of the controller algorithm without having to change any of the hardware.

The PID is perhaps the most common algorithm in use for system control, and there are many different forms used. The algorithm used here is commonly referred to as the position form of the discrete PID controller, and is given by

\[ m_n = K_c [e_n + (T/T_i) \sum_{k=0}^{n} e_k + (T_i/T)(e_n - e_{n-1})] + M_R , \]

where \( K_c \) is the proportional gain, \( T \) is the system sampling time, \( T_i \) is the integration (sometimes called reset) time, \( T_d \) is the derivative action time, \( M_R \) is the initial "valve" position (in this case the initial voltage squared applied to the heater), \( e_n \) is the error in the measured variable (in this case the block temperature is the measured variable; therefore, the error \( e_n \) is given by

\[ e_n = T_n - T_{ref} , \]

where \( T_n \) is the currently measured block temperature, and \( T_{ref} \) is the temperature at which the block is to be stabilized), and \( m_n \) is the new value for the manipulated variable (in this case it is the new "power" level, the voltage squared, to be applied to the block heater). It is referred to as the position form of the PID algorithm,
since it calculates the actual value of the manipulated variable each time, as opposed to just calculating a change in value of the manipulated variable. The latter form for the PID controller is commonly referred to as the velocity form.

One of the main problems of the position form for the PID controller is that it suffers from integration "windup". This refers to the fact that if a significant error occurs, the integral term can rapidly buildup to large values. This by itself is not important, since the control algorithm in the theoretical, non-physical form would calculate the necessary correction to the manipulated variable. But when it is combined with the fact that the manipulated variable is a real world quantity that has upper and lower limits on how far it can be manipulated (e.g. one cannot have negative power applied to the block heater, nor can one exceed the maximum output voltage of the heater power supply), this problem becomes disastrous. The controlled variable usually ends up saturated and system control breaks down. The classic symptom of integrator windup in a PID controller is an extreme overshoot of the set-point temperature, followed by a lengthy decay to the set-point. This symptom was observed many times in the initial development stages of the block heater algorithm before it was recognized as integral windup.

In order to correct this problem, the value of the manipulated variable (the block heater voltage squared) is sensed, and whenever this value exceeds the maximum output voltage squared of the DAC, the error sum is decremented by the amount that the (calculated) manipulated variable exceeds this maximum value. Since the block heater voltage is controlled via DAC 0 on the TECMAR Labmaster board (which in turn programs the Harrison programmable power supply), and is set
for a +/- 10.0 VDC range; this in effect turns off the integral action whenever the "power" level exceeds 100 volts². This allows the system to recover faster than would otherwise be allowed. The incorporation of this feature into the control algorithm is usually referred to as "limited-integral-action" or "anti-reset-windup."

Another problem with the PID controller is that it must be "tuned" to the system that it will be controlling, since this algorithm is a general purpose control algorithm. Again, there are many different possible tuning procedures, but the procedure used for tuning this system was that of C. L. Smith, and is strictly empirical. Essentially, this can be described as setting the proportional gain to 1, setting \( T/T_i \) and \( T_d \) equal to zero, and examination of the system response to a step input of the heater voltage. The plot of the system response is then examined for a system dead-time, and a response slope that is then compared to a chart of values. Furthermore, the system proportional gain is increased steadily until the system breaks into oscillation, and this maximum value for \( K_c \) is the final parameter used in the chart in order to set "optimal" values for \( K_c \), \( T_i \), and \( T_d \).

This tuning method relied on the fact that the system response remains the same from run to run, which rarely occurred in practice for this probe. The system heat leaks, thermal time constants, block mass, etc. changed from run to run as well as having a significant temperature dependence. For example, the block-to-\(^4\)He thermal link was strongly dependent on the \(^3\)He vapor pressure in the \(^3\)He pumping line, and above 4.2K there was usually \(^4\)He in this pumping line, which has different thermodynamic properties. However, it usually was necessary to only change the proportional gain of the system (gain scheduling), and occasionally the
integration time, to accommodate these changing system parameters.

With the appropriate PID parameters set, and the $^3$He refrigerator active, temperature stabilization on the order of 2.0 millikelvin was typical for block temperatures in the range of 0.5K to 7.0K. This was with a sampling rate of 0.2 seconds for the block control algorithm. With the addition of a system model for the probe, and sampling rates below 0.1 seconds, one could expect temperature noise due to the control algorithm to be reduced below 1.0 millikelvin for block temperatures in the range of 0.5 to 25 Kelvin. This is one obvious area for future work.

3. Thermometer Uncertainties

In order to evaluate the uncertainties and errors associated with each of the thermometers used in the data collection system, the sensitivity functions for each thermometer must be evaluated. These functions are a measure of the ratio of the fractional change of resistance to the fractional change in temperature, as a function of temperature. To arrive at the calculated sensitivity function for a given thermometer, the following function must be evaluated:

$$\text{Sensitivity} = S(T) = \frac{d(\log_e(R))}{d(\log_e(T))}.$$ 

For the 1k-ohm chip resistors, if one uses the previously given temperature fitting function, the sensitivity function fits the form

$$S_{1k-\text{ohm}}(T) = (-1/4)\sum_{i=1}^{N} iA_i T^{-4/m}.$$ 

For the high temperature germanium thermometer, this function is of the form
For the low temperature germanium thermometer, this function is of the form

$$S_{\text{low-germ}}(T) = B_{-} T^{-1} - \sum_{i=1}^{12} B_{i} T^{i}.$$ 

These calculated sensitivity functions for the germanium thermometers are plotted versus temperature in Figure 24 and Figure 25.

In order to arrive at an estimate of the absolute temperature error of the high temperature germanium thermometer, for example, one uses the sensitivity function of Figure 25. At 15 Kelvin, the sensitivity function is approximately 2.0; therefore, the fractional uncertainty of the temperature is approximately 0.5 times the fractional uncertainty of the resistance. Since the fractional uncertainty of the resistance is equal to the fractional uncertainty of the conductance for both of the germanium thermometers, the fractional uncertainty of the temperature is 0.5 times the fractional uncertainty of the conductance, i.e. at 15 K,

$$\frac{dT}{T} = 0.5 \times \frac{dG}{G}.$$ 

Thus, if there is an inherent error of 1% in the S.H.E. conductance value, then there is an inherent error of 0.5% in the temperature of the high temperature germanium thermometer at 15 K, i.e. the temperature is 15.0 +/- 0.075K. The actual error in the S.H.E. conductance value is a rather complicated function of conductance scale and conductance value; however, the bridge has been calibrated against a 0.01% General Radio Co. decade resistance box, model 1434-G, so that a reasonable estimate of the uncertainty in the overall conductance value is 0.5% of full scale. The conductance value is output as an analog voltage out of the S.H.E., and sent to
Figure 24
the analog to digital converters on the Tecmar board. The error associated with the Tecmar 16-bit ADC's is much less than 0.5% (typically .025%); therefore, the total system uncertainty of the conductance value can be estimated to be 0.5%. Looking at the overall sensitivity of the high temperature thermometer, one can set a gross overall sensitivity of 2.0, thus giving a gross overall temperature uncertainty of 0.25%.

One may initially be tempted to say that the sensitivity of the high-temperature germanium thermometer improves at temperatures lower than 1.2K, and this is true from interpretation of figure 25. Thus, one might say that this thermometer should be used below 1.2K, giving higher temperature accuracy and greater sensitivity for a given conductance value. This would indeed be the case if the instrumentation were perfect. However, there is a lower limit on the conductance value that can be accurately measured by the S.H.E. conductance bridge, typically about 0.020 mmhos (approximately 50k-ohms equivalent resistance). Figure 26 shows a plot of sensor resistance versus temperature for the high-temperature thermometer, and clearly shows that the sensor resistance becomes prohibitively high below 1.2K, necessitating the use of another block thermometer.

A similar analysis of the sensitivity of the low temperature germanium thermometer reveals that it is not particularly sensitive in the 0.5 to 5K range, exhibiting a gross overall sensitivity of 0.2, giving an overall temperature uncertainty of approximately 2.5%.

For the sample thermometers, the errors in the digital LIA and the ADC's must be taken into account in order to get an estimate of the sample thermometer
uncertainty. In addition, the sensitivity function of the sample thermometers also
should be evaluated. In order to determine this uncertainty, one needs to look at
the method of measurement of the resistance of the thermometer.

The bolometer resistance is measured with the LIA by applying a constant
amplitude A.C. current to the chip thermometer. The LIA picks off the A.C.
voltage across the thermometer, and the chip resistance is arrived at by the
following expressions

\[ V_{\text{bol}} = [0.1(V_1 - V_{\text{OFF}})/G_1 + V_{\text{LIA-offset}}]G_{\text{LIA}} , \]

and

\[ R_{\text{bol}} = V_{\text{bol}}R_{\text{series}}/(V_{\text{LIA-exc}} - V_{\text{bol}}) , \]

where \( V_1 \) is the voltage measured by ADC channel one; i.e., the output voltage of
the LIA in the range +/- 10 vdc, \( G_1 \) is the gain of channel one, \( V_{\text{OFF}} \) is the voltage
offset of channel one, \( G_{\text{LIA}} \) is the gain setting of the LIA, \( V_{\text{LIA-offset}} \) is the scale offset
of the LIA, \( R_{\text{series}} \) is a resistor in series with the bolometer resistor that has a much
higher resistance than that of the bolometer resistor (the "constant" current source
limiting resistor), and \( V_{\text{LIA-exc}} \) is the LIA excitation voltage applied across \( R_{\text{series}} \) and
\( R_{\text{bol}} \) in order to generate the constant current source.

In normal operation, the voltage across the bolometer is on the order of a
few millivolts, while the excitation voltage is usually on the order of 0.2 volts. For
the purposes of error analysis one can therefore approximate

\[ V_{\text{LIA-exc}} - V_{\text{bol}} = V_{\text{LIA-exc}} . \]

This reduces the expression for \( R_{\text{bol}} \) to

\[ R_{\text{bol}} = R_{\text{series}} V_{\text{bol}}/V_{\text{LIA-exc}} . \]
Figure 25
This gives an expression for the bolometer resistance that is the product of a series resistance times a function of the LIA. The stated system accuracy of the P.A.R. LIA is 1%, while the series resistor used had a stated and measured accuracy of 0.05%. This gives an accuracy for $R_{bol}$ of approximately 1.4%, i.e. the fractional uncertainty of $R_{bol}$ is 0.014.

For the 1k-chip bolometer, an expression for the uncertainty in temperature is therefore

$$dT/T = -4(0.014)/[\sum_i A_i T^{-\alpha_i}]$$.

For one of the 1k-chip resistor used in this experiment, this reduces to

$$dT/T = -4(0.014)/(1.495*T^{-0.25}) = -0.0375*T^{0.25}$$

The above relation indicates that the fractional uncertainty in temperature varies from approximately 3% at 0.5K to 5.5% at 5.0K. Assign an overall fractional uncertainty in the temperature for the 1k-chip thermometer of 0.04.

For the 100k-chip bolometer, a similar expression for the uncertainty in temperature is

$$dT/T = -(1/X)(0.014)/[\sum_i A_i T^{-\alpha_i}]$$.

For one of the 100k-chip resistor used in this experiment, this reduces to

$$dT/T = -0.00669*T^{0.76}$$

The above relation indicates that the fractional uncertainty in temperature varies from approximately 1.1% at 2.0K to 6.7% at 20K. Assign an overall fractional uncertainty in the temperature for the 100k-chip thermometer of 0.03.
Figure 26
4. Temperature Derivative Uncertainty

During the data analysis of the decay curves, it is necessary to evaluate the time derivative of this curve. A standard difference equation approach would result in an incredibly noisy derivative, since the slopes normally have rather small negative values (a typical value is approximately -0.005 Kelvin/second), and any noise in the temperature measurements could easily result in a positive slope, which is physically not realizable. Therefore, it is standard practice to fit the decay curve to a smooth polynomial and analytically determine the derivative. Two approaches are possible: cubic spline fits, and least-squares fits. Both methods were employed, with a smoothing cubic spline giving the "best" results, in terms of aesthetically pleasing, while least-squares produced results more amenable to analysis.

In order to analyze the uncertainties associated with LS analysis, consider the polynomial approximation given by

\[ \dot{y}(x_i) = a_1 + a_2x_i + a_3x_i^2 + \ldots + a_mx_i^m. \]

The LS procedure has to do with minimizing the following function with respect to the M+1 fitting parameters \( \alpha_k \)

\[ \chi^2 = \sum_{i=1}^{N} \left( \frac{y_i - \dot{y}(x_i)}{\sigma_i} \right)^2 \]

\[ = \sum_{i=1}^{N} \left( \frac{(y_i - \sum_{j=1}^{M} a_jx_i^j)/\sigma_i}{\sigma_i} \right)^2, \]

where \( \sigma_i \) is the uncertainty in the \( y_i \) value of the i-th data point. A differentiation of the above equation with respect to \( \alpha_k \) generates the following M+1 equations

\[ 0 = \sum_{i=1}^{N} [(y_i - \dot{y}(x_i))/\sigma_i](-2i\frac{d\dot{y}(x_i)}{d\alpha_k}). \]

If the equation is rewritten by replacing the expression for the approximating function \( \dot{y}(x_i) \), then the following M+1 equations result
\[ 0 = \sum_{i=1}^{N} \left( (y_i^k - \sum_{j=1}^{M} a_j x_i^{k-j}) / \sigma_i^2 \right). \]

With the definitions
\[ \beta_k = \sum_{i=1}^{N} (y_i^{k+1}) / \sigma_i^2 = S_k^{k+1}, \]
\[ \alpha_{kj} = \sum_{i=1}^{N} (x_i^{k+1} x_i^{j-1}) / \sigma_i^2 = S_k^{k+j-2}, \]
then the LS problem reduces to solving the matrix equation
\[ \beta = \alpha \cdot a \]

where 'a' is the M+1 by 1 matrix of fitting parameters. It explicitly yields the following equation
\[
\begin{pmatrix}
S_1^0 \\
S_1^1 \\
S_1^2 \\
\vdots \\
S_1^m
\end{pmatrix}
= 
\begin{pmatrix}
S_0^0 & S_0^1 & S_0^2 & \ldots & S_0^m \\
S_0^1 & S_0^2 & S_0^3 & \ldots & S_0^{m+1} \\
S_0^2 & S_0^3 & S_0^4 & \ldots & S_0^{m+2} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
S_0^m & S_0^{m+1} & \ldots & S_0^{2m}
\end{pmatrix}
\begin{pmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_m
\end{pmatrix}
\]

Then the solution of the system of equations can be obtained by calculating the inverse of the (m+1) by (m+1) matrix \( \alpha \). This solution can be formally stated as
\[ a = \alpha^{-1} \cdot \beta. \]

The uncertainties in the fitting parameters \( a_k \) can be obtained by the expression
\[ \sigma^2(a_k) = [\alpha^{-1}]_{kk}. \]

The first attempt at modelling a section of the decay curve in order to calculate the time derivatives was to apply a linear model
\[ \hat{y}(x_i) = a_0 + a_1 x_i. \]
However, depending on the number of data points used in the model, this gives only a crude approximation to the slope for data that has an obviously non-linear decay. This indicated the need for a higher order model.

Consider a second order approximation, the simple quadratic model given by

\[ \hat{y}(x_i) = a_0 + a_1 x_i + a_2 x_i^2 . \]

The expression for \( \beta = \alpha \cdot a \) simplifies to

\[
\begin{pmatrix}
S_{10}^{1,0} \\
S_{1,1}^{1,1} \\
S_{1,2}^{1,2}
\end{pmatrix} =
\begin{pmatrix}
S_{0,0}^{0} & S_{0,1}^{0} & S_{0,2}^{0} \\
S_{0,1}^{0} & S_{0,2}^{0} & S_{0,3}^{0} \\
S_{0,2}^{0} & S_{0,3}^{0} & S_{0,4}^{0}
\end{pmatrix}
\begin{pmatrix}
a_0 \\
a_1 \\
a_2
\end{pmatrix} .
\]

This expression is solved by inverting the matrix and multiplying by the inverse to give the solution vector

\[
a = \Delta^{-1} \begin{pmatrix}
+ S_{1,0}^{1,0} (S_{0,2}^{0} S_{0,4}^{0} - S_{0,3}^{0} S_{0,5}^{0}) - S_{1,1}^{1,1} (S_{0,1}^{0} S_{0,4}^{0} - S_{0,2}^{0} S_{0,5}^{0}) + S_{1,2}^{1,2} (S_{0,1}^{0} S_{0,3}^{0} - S_{0,2}^{0} S_{0,4}^{0}) \\
- S_{1,0}^{1,0} (S_{0,1}^{0} S_{0,4}^{0} - S_{0,3}^{0} S_{0,2}^{0}) + S_{1,1}^{1,1} (S_{0,0}^{0} S_{0,4}^{0} - S_{0,2}^{0} S_{0,3}^{0}) - S_{1,2}^{1,2} (S_{0,0}^{0} S_{0,3}^{0} - S_{0,2}^{0} S_{0,1}^{0}) \\
+ S_{1,0}^{1,0} (S_{0,0}^{0} S_{0,3}^{0} - S_{0,2}^{0} S_{0,1}^{0}) - S_{1,1}^{1,1} (S_{0,0}^{0} S_{0,3}^{0} - S_{0,1}^{0} S_{0,2}^{0}) + S_{1,2}^{1,2} (S_{0,0}^{0} S_{0,2}^{0} - S_{0,1}^{0} S_{0,1}^{0})
\end{pmatrix}
\]

where \( \Delta \) is the expression for the determinant of the matrix \( \alpha \), given by

\[ \Delta = \det(\alpha) = S_{0,0}^{0} (S_{0,2}^{0} S_{0,4}^{0} - S_{0,3}^{0} S_{0,5}^{0}) - S_{0,1}^{0} (S_{0,1}^{0} S_{0,4}^{0} - S_{0,2}^{0} S_{0,5}^{0}) + S_{0,2}^{0} (S_{0,1}^{0} S_{0,3}^{0} - S_{0,2}^{0} S_{0,4}^{0}) . \]

In addition, the equation for the uncertainties in the fitting parameters \( a_k \) can be expressed by

\[
\begin{pmatrix}
\sigma^2(a_0) \\
\sigma^2(a_1) \\
\sigma^2(a_2)
\end{pmatrix} = \Delta^{-1} \begin{pmatrix}
S_{0,2}^{0} S_{0,4}^{0} S_{0,3}^{0} \\
S_{0,0}^{0} S_{0,4}^{0} S_{0,2}^{0} \\
S_{0,0}^{0} S_{0,2}^{0} S_{0,1}^{0}
\end{pmatrix} .
\]

The above expressions are used for the fitting parameters and for their uncertainties. One can obtain a quadratic fit 'N' points for noisy data, and then calculate the
derivative analytically to determine the slope. The expression for the slope $M_i$ would of course be

$$\dot{y}(x_i) = M_i = a_1 + 2a_2 x_i.$$  

One estimate for the uncertainty in the slope can be derived by using the standard Gaussian distribution rules for uncorrelated uncertainty calculations of sums and products, i.e.

$$F = X + Y \text{ implies } \sigma_F^2 = \sigma_X^2 + \sigma_Y^2,$$

and

$$F = X^a Y^b Z^c \text{ implies }$$

$$\left(\frac{\sigma_F}{F}\right)^2 = \left(\frac{a\sigma_X}{X^a} + b\sigma_Y/Y^b + c\sigma_Z/Z^c\right)^2.$$  

Thus, the uncertainty in the slope for the quadratic model is given by

$$\sigma^2(M_i) = \sigma^2(a_i) + 4(X_i \sigma^2(a_2) + a_2^2 \sigma^2(X_i)).$$

For the analysis that follows, the uncertainty term containing $\sigma(X_i)$ will be assumed to be zero. This is not a totally unreasonable assumption, since this variable is the time in seconds of the decay curve. The absolute accuracy of this number is $1 \times 10^4$ seconds, typical values for $X_i$ are on the order of 100 seconds, and typical values for $a_2$ are on the order of $10^{-7}$/sec$^2$. For future reference, however, the early versions of the data collection program produced data that seemed to have an inordinately high noise in time. This was traced to the use of single precision arithmetic for the calculation of the time. As a result, during long time decay runs when times went over 10,000 seconds and sampling times were on the order of 1 point/sec, the floating point accuracy of the temporal position was reduced, inducing random time noise on the order of 0.01 seconds.
In addition to the observed time noise, it was also observed that a least-squares polynomial fit to the temperature versus time curve had some sensitivity to the starting time 't₀', which is contrary to intuition. This was traced to the fact that the starting times were usually several thousand seconds, which showed up as a rather ill-conditioned set of LS equations. For example, with a starting time of 2000 seconds, S°₀ would be of order N, while S°₄ would be of order N*10¹⁴. In order to correct this (in addition to performing double precision arithmetic), the starting time was subtracted from all subsequent times (coordinate scaling was not deemed necessary at this time). This was done, and analysis of the LS equations was performed to ensure that there was no hidden dependence on the starting time 't₀'.

Consider the coordinate change xᵢ = xᵢ + t₀. First of all, the individual elements of the LS matrix need to be inspected to determine how they transform with respect to this coordinate change. For the individual sums (k > 0),

\[
\frac{\partial}{\partial t₀} S^k = \frac{\partial}{\partial t₀} \sum_{m=1}^{N} \frac{[y'(xᵢ + t₀)]^k}{\sigmaᵢ^2}  \\
= k \sum_{m=1}^{N} \frac{y'(xᵢ + t₀)^{k-1}}{\sigmaᵢ^2}  \\
= k S^{k-1}
\]

Of course, the derivative of the k = 0 term is simply 0. Consider the determinant of the LS matrix,
\[
\frac{\partial}{\partial t_0} \Delta = 0 + S^{00}[2S^{01}S^{04} + 4S^{02}S^{03} - 2S^{01}(3S^{00})]
- S^{00}(S^{01}S^{04} - S^{02}S^{03}) - S^{01}[S^{00}S^{04} + 4S^{01}S^{03} - 2S^{01}S^{03} - 3S^{02}S^{01}]
+ 2S^{03}(S^{01}S^{03} - S^{02}S^{03}) + S^{02}[S^{00}S^{03} + 3S^{01}S^{02} - 2S^{03}(2S^{01})].
\]

A collection of terms in this expression indicates that they all cancel, leaving
\[
\frac{\partial}{\partial t_0} \Delta = 0.
\]

Now the derivatives of the fitting parameters can be evaluated easily. For \(a_2\),
\[
\frac{\partial}{\partial t_0} a_2 = \Delta^{-1} \frac{\partial}{\partial t_0} \left\{ S^{10}(S^{00}S^{03} - S^{02}S^{04}) \right\}
- S^{10}(S^{00}S^{03} - S^{02}S^{04}) + S^{12}(S^{00}S^{02} - S^{01}S^{01})
= \Delta^{-1} \left\{ S^{10}[S^{00}S^{03} + 3S^{01}S^{02} - 2S^{02}(2S^{01})] \right\}
- S^{10}(S^{00}S^{03} - S^{01}S^{02}) - S^{12}[3S^{00}S^{02} - S^{01}S^{04} - 2S^{01}S^{01}]
+ 2S^{14}(S^{00}S^{02} - S^{01}S^{04}) + S^{12}[2S^{00}S^{01} - 2S^{01}S^{04}]
= 0.
\]

This was expected, since \(a_2\) corresponds to 1/2 of the second derivative of the model, and the second derivative should not depend on the starting position of the x-variable. Evaluating the derivative of \(a_1\), one obtains
\[
\frac{\partial}{\partial t_0} a_1 = \Delta^{-1} \frac{\partial}{\partial t_0} \left\{ -S^{10}(S^{01}S^{04} - S^{03}S^{02}) \right\}
+ S^{10}(S^{00}S^{04} - S^{02}S^{02}), S^{12}(S^{00}S^{03} - S^{02}S^{01})
= \Delta^{-1} \left\{ 0 - S^{10}[S^{00}S^{04} + 4S^{01}S^{03} - 3S^{02}S^{02} - 3S^{03}S^{01}] \right\}
+ S^{10}(S^{00}S^{04} - S^{02}S^{02}) + S^{12}[4S^{00}S^{03} - 2S^{02}(2S^{01})]
- 2S^{14}(S^{00}S^{01} - S^{02}S^{01}) - S^{12}[3S^{00}S^{02} - 2S^{01}S^{01} - S^{02}S^{01}]
= \Delta^{-1} \left\{ -2S^{10}(S^{00}S^{03} - S^{02}S^{02}) + 2S^{11}(S^{00}S^{03} - S^{01}S^{02})
- 2S^{12}(S^{00}S^{02} - S^{01}S^{01}) \right\}
= -2a_2.
\]
Note that the derivative of \( a_i \) does not equal zero. This is not too surprising, since it has already been stated that \( a_i \) is not equal to the slope. It is expected, however, that the derivative of the slope with respect to starting position should be zero. In order to verify this, take the derivative of the slope with respect to \( t_0 \).

\[
\frac{\partial}{\partial t_0} M_i = \frac{\partial}{\partial t_0} [a_i + 2(x_i + t_0)a_2] = -2a_2 + 2a_2 = 0
\]

For completeness, the derivative of \( a_0 \) was also calculated, although this is expected to depend heavily on \( t_0 \).

\[
\frac{\partial}{\partial t_0} a_0 = \Delta^1 \frac{\partial}{\partial t_0} \{ S^{10}(S^{02}S^{04} - S^{03}S^{03}) 
- S^{11}(S^{01}S^{04} - S^{02}S^{03}) + S^{12}(S^{01}S^{03} - S^{02}S^{02}) \}
= \Delta^1 \{ 0 + S^{10}[2S^{01}S^{04} + 4S^{02}S^{03} - 2S^{03}S^{02}] 
- S^{11}[S^{00}S^{04} + 4S^{01}S^{03} - 2S^{01}S^{03} - 3S^{02}S^{02}] 
+ 2S^{11}(S^{01}S^{03} - S^{02}S^{02}) + S^{12}[S^{00}S^{03} + 3S^{01}S^{02} - 2S^{02}(2S^{01})] \}
= \Delta^1 \{ S^{10}(S^{01}S^{04} - S^{02}S^{03}) - S^{11}(S^{00}S^{04} - S^{01}S^{02}) 
+ S^{12}(S^{00}S^{03} - S^{01}S^{03}) \}
= -a_1.
\]

The derivative of \( a_0 \) does not vanish, but this particular result was not expected. On closer examination this result allows the model to be independent of starting \( x \) values. This can be verified by taking the derivative of the model equation as follows:

\[
\frac{\partial}{\partial t_0} \hat{y}(x_i) = \frac{\partial}{\partial t_0} [a_0 + a_1(x_i + t_0) + a_2(x_i + t_0)^2]
= -a_1 + (-2a_2)(x_i + t_0) + a_1 + 0 + a_2[2(x_i + t_0)]
= 0.
\]
The net results of these calculations are that the model, the first derivative of the model, and the second derivative of the model are all independent of starting value \( t_0 \), which agrees with intuition. Therefore, any dependence of the LS fitting upon the starting value \( t_0 \) must be some form of numerical instability.

In order to estimate the uncertainties in the model slope, several approximations must be made. First, it will be assumed that the time variable starts from zero. The analysis above indicated that this should not be a problem. Second, it will be assumed that the time values are equally spaced, so that for any of the 'N' data points the x-variable can be expressed as

\[ x_k = k \delta \]

where \( \delta \) is the temporal spacing of the data points. This is not far from the truth. However, the data collection routines are not interrupt-driven at this time; therefore, software polling was used to determine if it was time to collect another data point, resulting in a slightly uneven spacing in the time values. Third, it will be assumed, as stated earlier, that the uncertainty in the time values will be zero. Fourth, it will be assumed that the uncertainty in the temperature values will be the same over the entire range of the decay curve; i.e.,

\[ \sigma(T_k) = \sigma. \]

One possible way to determine this quantity \( \sigma \) is to use the residual sum of squares; i.e.,

\[ \sigma = \frac{\sum_{i=1}^{N} (y_i - \hat{y}(x_i))^2}{N}. \]

Fifth, it will be assumed that system accuracy is equal to system uncertainty, and that systematic errors will be treated as uncertainties. This means, for example, that
a systematic 1% error in the output voltage of the LIA will be treated as a 1% uncertainty in the LIA voltage. Armed with these approximations, the estimated uncertainties in the slopes of the decay curve can now be calculated.

If one uses the equally spaced data point assumption, along with the uniform uncertainty assumption, a closed form expression for the $S^{0k}$ sums can be obtained by using standard expressions for the sums of powers of integers.$^{24}$

\[
S^{00} = \sum_{i=1}^{N} 1/\sigma_i^2 = N/\sigma^2
\]

\[
S^{01} = \sum_{i=1}^{N} x_i/\sigma_i^2 = \sum_{i=1}^{N} (\delta i)^1/\sigma_i^2 = \delta N(N+1)/(2\sigma^2)
\]

\[
S^{02} = \sum_{i=1}^{N} x_i^2/\sigma_i^2 = \sum_{i=1}^{N} (\delta i)^2/\sigma_i^2 = \delta^2 N(N+1)(2N+1)/(6\sigma^2)
\]

\[
S^{03} = \sum_{i=1}^{N} x_i^3/\sigma_i^2 = \sum_{i=1}^{N} (\delta i)^3/\sigma_i^2 = \delta^3 N^2(N+1)^2/(4\sigma^2)
\]

\[
S^{04} = \sum_{i=1}^{N} x_i^4/\sigma_i^2 = \sum_{i=1}^{N} (\delta i)^4/\sigma_i^2 = \delta^4 N(N+1)(2N+1)(3N^2+3N-1)/(30\sigma^2)
\]

These values are placed into the determinant of the LS matrix to yield

\[
\Delta = S^{00}(S^{02}S^{04}-S^{03}S^{03})-S^{01}(S^{01}S^{04}-S^{02}S^{03})+S^{02}(S^{01}S^{03}-S^{02}S^{02})
\]

\[
= (\delta^4/\sigma^4) \{N[(N/2)(N+1)(N/30)(N+1)(2N+1)(3N^2+3N-1)-(N^2(N+1)^2)/16]
- (N/2)(N+1)[(N/2)(N+1)(N/30)(N+1)(2N+1)(3N^2+3N-1)]
- (N/6)(N+1)(2N+1)(N/4)(N+1)^2
+ (N/6)(N+1)(2N+1)[(N/2)(N+1)(N/4)(N+1)^2]
- (N/6)(N+1)(2N+1)^2 \}
\]

And after much painful arithmetic, this simplifies to

\[
\Delta = (\delta^4/\sigma^4)(N-2)(N-1)^2N^3(N+1)^2(N+2)/2160
\].
The expressions for the uncertainty in the LS fitting parameters can now be approximated.

\[ \sigma^2(a_c) = \Delta^{-1}(S_{00}S_{02}-S_{01}S_{01}) \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)[N(N/6)(N+1)(2N+1) -(N/2)^2(N+1)^2] \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)(N-1)N^2(N+1)/12 \]
\[ = 180(\sigma^2/\delta^2)/[(N-2)(N-1)N(N+1)(N+2)] . \]

\[ \sigma^2(a_o) = \Delta^{-1}(S_{00}S_{04}-S_{02}S_{02}) \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)[N(N/30)(N+1)(2N+1)(3N^2+3N-1) - (N/6)(N+1)(2N+1)^2] \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)(N-1)N^2(N+1)(2N+1)(8N+11)/180 \]
\[ = 12(\sigma^2/\delta^2)[(2N+1)(8N+11)]/[(N-2)(N-1)N(N+1)(N+2)] . \]

\[ \sigma^2(a_d) = \Delta^{-1}(S_{02}S_{04}-S_{03}S_{03}) \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)[(N/6)(N+1)(2N+1)(N/30)(N+1)(2N+1)(3N^2+3N-1) - (N/4)(N+1)^2] \]
\[ = \Delta^{-1}(\delta^2/\sigma^2)(N-1)N^2(N+1)(N+2)(3N^2+3N+2)/720 \]
\[ = 3\sigma^2(3N^2+3N+2)/[(N-2)(N-1)N] . \]

From these values the uncertainty in the LS slope for the quadratic model can be determined.

\[ \sigma^2(M_o) = \sigma^2(a_c) + 4X_0^2\sigma^2(a_o) . \]

If the slope is evaluated at the midpoint of the 'N' data points, i.e. at \( x_{N/2} = \delta N/2 \), then the uncertainty becomes

\[ \sigma^2(M_{N/2}) = \sigma^2(a_c) + N^2\delta^2\sigma^2(a_o) \]
\[ = 12(\sigma^2/\delta^2)[(2N+1)(8N+11)]/[(N-2)(N-1)N(N+1)(N+2)] \]
\[ + 180N^2 \delta^2 (\sigma^2/\delta^4)/[(N-2)(N-1)N(N+1)(N+2)] \]

\[ = 12(\sigma^2/\delta^4)((2N+1)(2N+1)+15N^2)/[(N-2)(N-1)N(N+1)(N+2)] \]

\[ = 12(\sigma^2/\delta^4)(31N^2+30N+11)/[(N-2)(N-1)N(N+1)(N+2)] . \]

In order to interpret these results, consider only the leading order terms in \( N \); i.e., treat the polynomials in each of the above expressions as a constant times the leading power of \( N \). Then we have the following approximations:

\[ \sigma^2(a_2) = 180(\sigma^2/\delta^4)O(N^{-5}) = 180\sigma^2/[N(N\delta)^4] \]

\[ \sigma^2(a_1) = 12(\sigma^2/\delta^4)O(16N^2)/O(N^4) = 192\sigma^2/[N(N\delta)^2] \]

\[ \sigma^2(a_0) = 3\sigma^2O(3N^2)/O(N^3) = 9\sigma^2/N \]

\[ \sigma^2(M_{R\delta}) = 12(\sigma^2/\delta^4)O(31N^2)/O(N^3) = 372\sigma^2/[N(N\delta)^3] \].

Note that for increasing \( N \), this approximation becomes closer to the exact form. At \( N = 10 \), this approximation for the uncertainty is in error by -7\% (i.e., it is too small by 7\%), and at \( N = 50 \) it is in error by -1.1\%.

In this form, the quantity \( N\delta \) represents the time that the LS model is fit to the data. There is normally an upper limit on this number, a physical limit set by the time constants involved in the sample and probe, and not merely a whim of the experimenter.

For example, with \( N = 15 \) and \( \delta = 1 \) second, \( N\delta \) represents 15 seconds, the \( x \)-variable range over which the model is fit. Therefore, if one looks at the above approximations for the uncertainties, it becomes clear that the uncertainties all have an \( N^{-1/2} \) dependence, but each has a distinct inverse dependence on the modelling time. Note also that the uncertainty in \( a_0 \) has no dependence on the modelling time; it depends only on the square root of the total number of points used in the model,
as it should, since this represents an offset of the y-variable.

Take a particular example, for \( N = 21 \), \( \delta = 0.5 \) seconds, \( \sigma = 0.01 \) Kelvin, and slope \( M_{11} = 0.01 \) Kelvin/sec, the fractional uncertainty in the slope is given by

\[
\frac{\sigma(M_{10})}{M_{10}} = 19.3\sigma/[M_{10}\sqrt{N}(N\delta)]
\]

\[
= 19.3\times0.01K/[0.01K/S\times21^{1/2}\times10.5S] = 0.40 = 40\%
\]

This demonstrates very clearly that the time derivatives are very noisy. It also indicates that increasing the sampling rate is essential in order to reduce this uncertainty. This can be demonstrated by increasing the sampling rate by a factor of ten: i.e. setting \( \delta = 0.05 \) seconds. Maintain the same total time for the model, \( N\delta = 10.5 \) seconds, requires that the number of data points used must be increased to \( N = 210 \). Thus the fractional uncertainty in the slope becomes

\[
\frac{\sigma(M_{10})}{M_{10}} = 19.3\sigma/[M_{10}\sqrt{N}(N\delta)]
\]

\[
= 19.3\times0.01K/[0.01K/S\times210^{1/2}\times10.55S] = 0.12 = 12\% ,
\]

an improvement by a factor of \( 10^{1/2} \).

For example, in the zero-field pure-barium sample in this experiment, typical values for the statistical uncertainty, derived from the residual sum of squares in data file 'GD0707.0', in the temperature of the 1k-chip resistor at 1.0 Kelvin are on the order of .05\%, therefore \( \sigma = 0.0005K \). The interval over which the quadratic model was fitted was 25 seconds, with \( N = 25 \), and \( \delta = 1.0 \) seconds. The slope was approximately 0.0011 Kelvin/seconds. These values yield

\[
\frac{\sigma(M_{12})}{M_{12}} = 19.3\sigma/[M_{12}\sqrt{N}(N\delta)]
\]

\[
= 19.3\times0.0005K/[0.0011K/S\times25^{1/2}\times25S] = 0.070 = 7.0\% .
\]
Note that this value is NOT the accuracy of the time derivative measurement; rather it is the uncertainty in the measurement due to fluctuations in the temperature.

5. Power Calibration Uncertainties.

Each of the data points used in the power loss calibration have a recorded uncertainty that was measured while the system was coming to equilibrium at that power level. Normally, a simple quadratic model was applied to the power loss curve, with only three data points measured. Since a quadratic model has three unknowns, this means there is only a single solution to the LS equation, and it has zero uncertainty. Therefore, the uncertainty in the power fit is given by the maximum uncertainty in the measured power levels. Typical values for the fractional uncertainty in the temperatures at these power levels were on the order of $10^{-3}$. The accuracy of the power fit is directly proportional to the uncertainty (accuracy) of the temperature. That is,

$$P_{L}(T,T_0) = K(T^2 - T_0^2)$$

implies that

$$(\sigma_{p}/P)^2 = (\sigma_{K}/K)^2 + (2\sigma_{T}/T)^2$$

If a quadratic model is being used, then the fractional error in the power fit is approximately $2*\sigma_{T}/T$.


With the expression for the specific heat as follows,

$$c_{p}(T) = -[P_{L}(T,T_0)/(dT/dt)]\{(\text{Mol.Weight})/M_{\text{sample}}\}$$
the uncertainty in the specific heat can be expressed by

\[
\left( \frac{\sigma_c}{c_p} \right)^2 = \left( \frac{\sigma_p}{P} \right)^2 + \left( \frac{\sigma_{\text{Mass}}}{\text{Mass}} \right)^2 + \left( \frac{\sigma_{\text{fit}}}{T} \right)^2 .
\]

The sample mass, on the order of 0.010 grams, was measured on a Heusser analytical balance with an uncertainty of 0.00005 grams. Therefore, the fractional uncertainty in the sample mass is about 0.005 = 0.5%. From the above analysis, it is clear that the fractional uncertainty in the temperature derivative and the uncertainty in the power fit are by far the more dominant terms in this uncertainty relation. Therefore, the uncertainty in the specific heat can be approximated by setting it equal to the sum of squares of uncertainty in the time derivative and uncertainty in power fit, yielding

\[
\frac{\sigma_c}{c_p} \approx \left[ \left( \frac{\sigma_{\text{fit}}}{\Gamma} \right)^2 + \left( 2 \frac{\sigma_T}{T} \right)^2 \right]^{1/2} .
\]

Thus, for the example cited above with the uncertainty in the time derivative = 7%, the fractional uncertainty in the temperature is 3%; therefore, the uncertainty in the power fit is approximately 6%, and finally the fractional uncertainty (accuracy) of the specific heat is 10%.

The way that experimenters typically bypass, or get around this tedious error analysis is by measuring the specific heat of a known sample, such as copper, and comparing their results with the NBS results. When the specific heat of a 33mg, 0.9999 copper sample was measured with the above apparatus, the measured error in the specific heat was 12%, although this error did not take into account the specific heat of the addenda attached to the copper sample, which is on the same order of magnitude as the copper. As it turns out, measuring the specific heats of these magnetic materials virtually eliminates the need to worry about the specific heat of
the addenda, since this value is typically three orders of magnitude below that of the sample.
APPENDIX B
COLLECTION/COMPUTATION PROGRAMS

1. MAIN DATA COLLECTOR

The main data collection programs used throughout this experiment were known as the QP-series of data collection programs. Their purpose was to totally automate the data collection process for the measurement of specific heat of samples of small mass via the relaxation method.

The listings that follow for the main data collection program and for the main specific heat calculation program will be more closely spaced for readability and brevity.
The following is the main QP-series data collection module.
The Program outputs voltages to the system He3 block heater and
to the system sample heater via two 12-bit DAC’s and reads the
system response on up to 8 16-bit FPGA-ADC channels and the SHE
conductance bridge. The SHE conductance value is read in through
16 of the 24 digital I/O lines (ports A and B). Data will be stored
in a file to be determined at execution time.

The TECMAR board must be in the I/O mapped mode, and the
correct switch settings on the daughter board must match the
number of ADC channels in current use.
Gain settings are determined via software.

The modules necessary for running this program are:
QPNEW.BAS or SIMQPNEW.BAS (if simulating program)
TECMAR.BAS or STECMAR.BAS (if simulating program)
THERMOM.BAS
These are automatically loaded when activating Quick Basic and
loading the first module listed, as long as the appropriate
"*.MAK" file is present in the current working directory.

The following is the current ADC channel setup:

ADC(1)=LOCK IN AMPLIFIER OUTPUT (I.E. BOLOMETER VOLTAGE) Diff.In.
ADC(2)=STRAIN GAUGE HEATER VOLTAGE
ADC(3)=MAGNETIC FIELD 50 MILLI-VOLT SHUNT (S/C 6 TESLA MAGNET)
ADC(4)=ANALOG SHE OUTPUT (ON BLOCK THERMOMETER)
ADC(6)=FIXED-POINT CALIBRATION LIA VOLTAGE

The functions determined by these analog-to-digital values are
determined in the subroutine called 'Units Conversions, temperature
calculations, etc.'

The outputs to the system are via the 12-bit digital-to-analog
(DAC) converters, and the 24 digital I/O lines

DAC(0)=BLOCK TEMPERATURE CONTROL OUTPUT +/-10.0 VDC
DAC(1)=SAMPLE TEMPERATURE STEP OUTPUT +/- 2.5 VDC.

PORT A = SHE CONDUCTANCE BRIDGE
PORT B = SHE CONDUCTANCE BRIDGE
PORT C = NO CONNECTION
USAGE WITH PRINT SCREEN FUNCTION:

IF YOU PLAN ON USING THE PRINT-SCREEN FUNCTION TO PRINT THE
GRAPHICS SCREEN ON THE SYSTEM PRINTER, THEN YOU NEED TO SET UP
THE CORRECT VERSION OF THE PRINT-SCREEN TERMINATE-AND-STAY-
RESIDENT PROGRAM. THIS IS ACCOMPLISHED BY RUNNING THE PROGRAM
CALLED 'SCRNDMP.COM' BEFORE EXECUTING THE 'QP' PROGRAM. FOR
EXAMPLE, IF YOU ARE USING THE CGA GRAPHICS SCREEN, THEN YOU CAN
USE THE OLD 'GRAPHICS' COMMAND, OR YOU CAN USE THE 'SCRNDMP'
COMMAND AS FOLLOWS:

```
SCRNDMP /C /E
```

THIS TELLS THE 'SCRNDMP' ROUTINE TO USE CGA GRAPHICS DUMP TO AN
EPSON STYLE PRINTER. CONVERSELY, IF YOU WISH TO USE EGA GRAPHICS
MODES FOR THE DISPLAY SCREEN, THEN THE FOLLOWING SYNTAX IS USED:

```
SCRNDMP /E /E
```

THIS TELLS THE 'SCRNDMP' ROUTINE TO USE EGA GRAPHICS DUMP TO AN
EPSON STYLE PRINTER.
EITHER WAY, IN ORDER TO PRINT THE GRAPHICS SCREEN, YOU NEED TO
WAIT FOR A SYSTEM INPUT PROMPT, THEN PRESS THE 'PRINT-SCREEN'
KEY ON THE SYSTEM CONSOLE. IF NO INPUT IS GIVEN WITHIN TEN
SECONDS, THE TEXT SCREEN WILL BE PRINTED, NOT THE GRAPHICS SCREEN.
THEREFORE, WITHIN TEN SECONDS OF PRESSING THE 'PRINT-SCREEN' KEY
PRESS THE NUMERIC KEY FOR THE GRAPHICS SCREEN THAT YOU WANT TO
PRINT (PRESSING '1' WILL PRINT THE GRAPHICS SCREEN THAT IS
CURRENTLY DISPLAYED). OR, IF YOU JUST WANT TO PRINT THE TEXT
SCREEN, PRESS THE SPACE-BAR FOR INSTANT PRINTING OF THE TEXT
SCREEN.

DESCRIPTION OF INTERNAL (MAIN MODULE) SUBROUTINES

*statinit* -- this routine initializes (zeros) all of the
statistical sums used by the main module for
finding the average values and for determining
the zero slope condition

*statcalc* -- performs the modulo arithmetic needed for the
ring storage sums, and updates those sums

*unitsconversion* -- calculates the various physical values that are
measured by the ADC's. it also calculates the
bolometer temperature and the block temperature.
displayupdate -- plots the bolometer and block temperatures, determines if the user wants to stop the run, and also checks to see if the slope termination condition has been met.

maindataloop -- calls the ADC routine, gets the SHE value, calls the AMD9513 timer routine, calculates the units conversion, determines when it is time to call the PID controller, when to update the display, when to collect the next data point, and prints the time.

DESCRIPTION OF EXTERNAL SUBROUTINES:

sub getparms() -- calls the subroutine 'parminit' which sets up the parameters menu, and upon return it sets up almost all of the values in the COMMON SHARED block.

sub parminit() -- sets up the initial parameters menu, reads in the last set of parms and changes them according to the user's input. It then stores these new values in the parameters file.

sub tecinit() -- sets up the TECMAR starting address, and then initializes the SHE, the ADC's, and the AMD-9513 timer chip to count time in .1ms increments. It also zeros the variables 'reftime, contime, tim0#'

sub screeninit -- sets up the graphics display screen in CGA(EGA) and draws a box around it, displaying the X and Y axes RANGE/OFFSET information.

sub graph() -- plots the (X,Y) point onto the display screen in the color ICOLOR.

sub getadc() -- calls upon the TECMAR 16-bit analog to digital converters to collect up to 8 differential input, programmable gain channels.

sub dac() -- calls upon the TECMAR digital to analog converters to output an analog voltage to 1 of 2 DAC's.

sub amd tim() -- calls upon the TECMAR AMD-9513 timer chip to measure the time in .1 milli-second increments since the experiment began.
sub shepcb() -- calls upon the TECMAR digital I/O facilities to
get all of the switch settings and the digital
value of the conductance being measured by the
SHE potentiometric conductance bridge. It also
calculates the analog value of the conductance
that is output by the modified SHE bridge.

sub pidcontrol -- performs a (P)roportional-(I)ntegral-
action-(D)erivative control algorithm for the
purpose of controlling the block temperature via
measurements of the appropriate block thermometer.
it also prevents integral windup due to an
overload condition

sub powerfit() -- fits the bolometer temperature (squared) vs. the
the heater power input (via the strain gauge
sample heater) to a Least-squares line, and
outputs the predicted gauge resistance, power,
and the power fit parameters to the storage file.

sub lotempge() -- calculates the temperature of the low-temperature
germanium thermometer from the conductance value,
or it calculates the conductance value from the
temperature.

sub hitempge() -- calculates the temperature of the high-temperature
germanium thermometer from the conductance value,
or it calculates the conductance value from the
temperature.

sub block5k() -- calculates the temperature of the 5k Bi-Ruthenate
chip resistor from the conductance value, or it
calculates the conductance value from the temp.

sub genterm --> USE THIS SUBROUTINE FOR INSTALLATION OF A NEW
THERMOMETER!!
generic routine to calculate the temperature of
an arbitrary block thermometer given the
conductance value, or to calculate the conductance
value given the block temperature.

sub blocktemp() -- this routine calls the appropriate block temp
routine depending on which thermometer was
selected during 'parminit'

-----------------------------------------------------------------------

IF YOU CHANGE BOLUMETERS OR BLOCK THERMOMETERS, ALL OF THE
APPROPRIATE TEMPERATURE CONSTANTS WILL HAVE TO BE CORRECTED.

MODIFICATION HISTORY OF QP-SERIES OF DATA COLLECTOR PROGRAMS

2/12/89 --- QUICK BASIC 4.5 TRANSLATION

the entire QP series data collector was implemented
in Quick Basic 4.5, with separate modules for the
thermometry routines and for all routines using
the Tecmar board

3/20/89 --- DOUBLE PRECISION TIM# ROUTINE IMPLEMENTED

the timer routine that accesses the AMD-9513 needed
to be upgraded to double precision math, and all calls
to the timer routine 'AMDTIM' had to be made aware of
the double precision nature of 'TIM#'

3/25/89 --- PARMS INITIALIZATION ROUTINE MODIFIED

the parameter initialization and storage routine has
been modified, and order of storage of the constants
has been changed. the modifications were mostly
cosmetic.

4/02/89 --- TEMPERATURE ROUTINES MODIFIED

all of the block temperature routines were modified to
take advantage of the SELECT-CASE statement in Q-basic
and slight improvements were made to HITEMPGE to make it
approximately twice as fast.

DIM lgain(17), gain(17), OFFS(17), ADC(17)

' set up the ring storage space ---------------

DIM XTIM(1001), YLIA(1001), BOL(1001), BVOL(1001)
DIM RGVOL(1001), SHE(1001)

' common block ----------------

COMMON SHARED parmsfile$
COMMON SHARED tim#, reftime, contime
COMMON SHARED sheval, getemp, IFUNCT, lgain(), gain(), OFFS(), ADC()

COMMON SHARED ANSHE, ADCCCHAN, ADD, VOLTDAC, DACNUM
COMMON SHARED settemp, tint, tdiff, tsamp, kgain, mr, errsum
COMMON SHARED errblk, lsterr, therm, VOLT
COMMON SHARED LIGAIN, OFFSET, TC, LIAFRQ, EXVOLT, SERIESR, store$
COMMON SHARED VOLTI, VOLTF, VSTEP, REFRESH, PNTSEC, RGAUGE, HEATRES
COMMON SHARED offsetx, rangey, offsety, rangey, ACC, NPNT, FINC
COMMON SHARED scrx, scry, TSCRY, icolor
COMMON SHARED tim0#

' the following routines are contained in the module "QPNEW.BAS" which
' is identical with the module "SIMQPNEW.BAS" used in the simulator

' This is the main module

DECLARE SUB GETPARMS ()
DECLARE SUB SCREENINIT ()
DECLARE SUB GRAPH (x, Y, icolor)
DECLARE SUB PIDCONTROL ()
DECLARE SUB PARMINIT ()
DECLARE SUB POWERFIT (BOLTEMP!, VOLGAG!, VZERO!, powfunt)

'the following routines are contained in the Mod "TECMAR.BAS" or if we
' are using the simulation routines, they are in "STECMAR.BAS"

DECLARE SUB TE CINIT (ADD!, reftime!, contime!, tim0#)
DECLARE SUB amd tim (ADD!, tim#, tim0#)
DECLARE SUB SHEPCB (a!, A4!, G4!, O4!, LG4!, ANSHE!, sheval!, OVLD$)
DECLARE SUB DAC (ADD!, VOLTDAC!, DACNUM!)
DECLARE SUB GETADC (ADD!, ADCCHAN!, ADC(!), lgain())

'the following routines are contained in the module "THERMOM.BAS"

DECLARE SUB BLOCKTEMP (sheval!, getemp!, IFUNCT!, therm!)
DECLARE SUB GENTHERM (sheval!, getemp!, IFUNCT!)
DECLARE SUB LOTEMPGE (sheval!, getemp!, IFUNCT!)
DECLARE SUB HITEMPGE (sheval!, getemp!, IFUNCT!)
DECLARE SUB BLOCK5K (sheval!, getemp!, IFUNCT!)

'set up some basic constants used in most of the data collection runs

MAGCONST = 285!:
'n' magconst*volshunt = field in tesla
RGauge = 108.9 + .2:
'strain gauge+1/2 leads for .5K-10K
VZERO = 0
icolor = 2

'THE FOLLOWING SETS UP THE FITTING PARAMETERS FOR THE DIFFERENT
' BOLOMETERS.

SET BOLOMET=1 IS FOR THE 1K
SET BOLOMET=2 IS FOR THE 5K
SET BOLOMET=3 IS FOR THE 10K
SET BOLOMET=4 IS FOR THE 100K

BOLMIN IS THE MINIMUM VALUE ACCEPTABLE FOR THE CALIBRATION
BOLMAX IS THE MAXIMUM VALUE ACCEPTABLE FOR THE CALIBRATION
XEXP IS THE EXPONENT USED IN THE FITTING EQUATION
In(bolres) = A0 + A1/T^XEXP + A2/T^(2*XEXP) + ....

BOLOMET = 3: ' SET UP FOR 10K BOLOMETER

DIM a(4, 4), XEXP(4), BOLMIN(4), BOLMAX(4)

' read in the exponents and the minimum calibration resistances,
' i.e. the highest allowed temperatures.
' note that the bolmax() values have not been implemented

DATA .25, 1200!, .25, 7400!, .761, 12000!, .7678, 123000!
FOR i = 1 TO 4
  READ XEXP(i), BOLMIN(i)
NEXT i

DATA 6.0776#, .9.65263169#, .9.24573715#, 1.495#, -3.6687611#
DATA 2.021132286#, 2.7243744#, 0, 3.90541956#, 0, 0, -0.06022042#, 0, 0
FOR j = 1 TO 4
  FOR i = 1 TO 4
    READ a(i, j)
  NEXT i
NEXT j

'----- set initial guesses of bolometer and block temperatures ------
BOLTEMP = 1: BLKTEMP = 1

'----- initialize some of the PID controller values ---------------
errsum = 0: errblk = 0: lsterr = 0

'----- GET ALL OF THE INPUT PARAMETERS FOR THE RUN ---------------
CALL GETPARMS

'----- initialize the block therm constants-----------------------
IFUNCT = 2
CALL BLOCKTEMP(sheval, getemp, IFUNCT, therm)

'----- initialize SHE, AMD9513, and ADC
CALL TECINIT(ADD, reftime, contime, tim0#)
CALL amdtrim(ADD, tim#, tim0#): ' -------zero tim0#----------------

'--------draw the real-time display window ----------------------
CALL SCREENINIT: LOCATE 22, 1
PRINT " <RET> stops data collection") TIME ="
LOCATE 23, 41: PRINT "Bol(kohms)="
LOCATE 4, 5: PRINT " file="; store$
LOCATE 21, 41: PRINT "block temp="

'-------- initialize the power stat-sums------------------------
CALL POWERFIT(BOLTEMP, VOLGAG, VZERO, 0)
IFUNCT = 0: ' normal function for temp routine
BEGIN DATA COLLECTION

START THE BOLOMETER HEAT STEPS

FOR VOLT = VOLTI TO VOLTF STEP VSTEP
  GOSUB statinit;  ' INITIALIZE STAT REGISTERS -------
  LOCATE 23, 10: PRINT "SAMP HEAT VOLTAGE = "; VOLT; "  "
  CALL DAC(ADD, VOLT, 1);  ' OUTPUT VOLTAGE TO BOLOMETER

DO
  GOSUB maindataloop:  ' CALL MAIN DATA COLLECTION ROUTINE
    Y = BOLTEMP: x = tim#
  GOSUB statcalc:  ' CALCULATE THE STAT SUMS
  LOOP WHILE flat <> 1:  ' FLAT is returned from slope calc

  ----- EQUILIBRIUM REACHED (FLAT = 1). RECORD AVG FOR POWER CALCS. -----
  ----- don't record averages if not enough points have been averaged ----

  IF II >= NPNT + 1 THEN
    BOLTEMP = YSUM / NPNT
    BOLK = BSUM / NPNT
    sheval = SSUM / NPNT
    VOLBOL = VSUM / NPNT
    VOLGAG = RGSUM / NPNT - VZERO
    IF VOLT = 0 THEN
      VZERO = VOLGAG
    END IF

    -- NOTE THAT TIM IS NOT AVERAGED
    -- get the average block temperature from average SHEVAL
    CALL BLOCKTEMP(sheval, getemp, IFUNCT, therm)

    -- store the averaged values
    PRINT #1, " SAMPLE HEATER VOLTAGE = "; VOLT
    PRINT #1, USING ";;;;;;; "; tim#; BOLTEMP; BOLK;
    PRINT #1, USING ";;;;;;; "; VOLBOL;
    PRINT #1, USING ";;;;;;; "; getemp; sheval

    ' calculate the standard deviations of the bolometer temperature and
    ' the standard deviation of the SHE conductance values
    TDEV = SQR(ABS(YSUM2 - YSUM ^ 2 / NPNT) / NPNT)
    SHEDEV = SQR(ABS(SSUM2 - SSUM ^ 2 / NPNT) / NPNT)
    PRINT #1, " tempdev="; TDEV;
    PRINT #1, " shedev="; SHEDEV; " # of points="; NPNT

    -- update power sums and output power value to storage file
    CALL POWERFIT(BOLTEMP, VOLGAG, VZERO, 1)
  END IF
NEXT VOLT

'-----------------------------------------DONE WITH HEAT STEPS -----------------------------------------'

calculate the simple power fit and output results to the storage file

CALL POWERFIT(BOLTEMP, VOLGAG, VZERO, 2)

VOLT = 0
GOSUB statinit: ' INITIALIZE STAT REGISTERS ------
LOCATE 23, 10: PRINT "SAMP HEAT VOLTAGE = "; VOLT; " "

'------turn off heat to the bolometer -----------------------------

CALL DAC(ADD, 0, 1)

'----------------------------------BEGIN DATA COLLECTION ON DECAY CURVE----------------------------------

DO
GOSUB maindataloop: ' CALL MAIN DATA COLLECTION ROUTINE
Y = BOLTEMP; x = tim#
GOSUB statcalc: ' CALCULATE THE STAT-SUMS
'---record all of the temperatures, etc., until flat = 1 ---------
PRINT #1, USING "#####.##### "; tim#; BOLTEMP; BOLK;
PRINT #1, USING "*.#####^### "; VOLBOL;
PRINT #1, USING "######## "; getemp; sheval
LOOP WHILE flat <> 1

'----------------------------------END OF DATA COLLECTION ----------------------------------

stopitnow:

CLOSE #1
LOCATE 2, 2: PRINT " DATE = "; DATE$
LOCATE 3, 2: PRINT "END-TIME = "; TIME$
LOCATE 23, 1: PRINT "PRINT IT ";
LOCATE 23, 10: INPUT DUMMY$

'------------------- STOP THE BOLOMETER HEATER-------------------

CALL DAC(ADD, 0, 1)
LOCATE 22, 1: INPUT "SHUT OFF BLOCK HEATER (y/n)"; a$
IF UCASE$(a$) = "Y" OR a$ = "" THEN
  CALL DAC(ADD, 0, 0): ' STOP THE BLOCK HEATER
END IF
SCREEN 0
STOP
statistical sums initialization
zero the stat sums used for the ring storage space used to determine
the termination condition

VARIABLES USED/CHANGED — ISTAT, NPNT, YLIA(), XTIM(), SHE(), BOL(),
BVOL(), RGVOL(), SSUM, SSUM2, BSUM, VSUM, RGSUM,
XSUM, XSUM2, XYSUM, YSUM, YSUM2, II, FLAT

statinit:

FOR ISTAT = 1 TO NPNT + 1
  YLIA(ISTAT) = 0
  XTIM(ISTAT) = 0
  SHE(ISTAT) = 0
  BOL(ISTAT) = 0
  BVOL(ISTAT) = 0
  RGVOL(ISTAT) = 0
NEXT ISTAT
SSUM = 0: SSUM2 = 0: BSUM = 0: VSUM = 0: RGSUM = 0
XSUM = 0: XSUM2 = 0: XYSUM = 0: YSUM = 0: YSUM2 = 0
II = 0: flat = 0
RETURN

statistical sums calculation -- RING STORAGE CALCULATIONS

II — is the number of the actual data points currently collected
NPNT — is the size of the current ring storage space
PNT — gives the current pointer position in the ring storage space
LPNT — gives the pointer position in the ring storage space of the
point that is to be deleted from the ring

VARIABLES USED/CHANGED — ISTAT, NPNT, YLIA(), XTIM(), SHE(), BOL(),
BVOL(), RGVOL(), SSUM, SSUM2, BSUM, VSUM, RGSUM,
XSUM, XSUM2, XYSUM, YSUM, YSUM2, II,
PNT, LPNT, X, Y, SHEVAL, BOLK, VOLBOL, VOLGAG
statcalc:

II = II + 1
PNT = (II MOD (NPNT + 1)) + 1
LPNT = ((II + 1) MOD (NPNT + 1)) + 1
YLIA(PNT) = Y: XTIM(PNT) = x
XSUM = XSUM + x - XTIM(LPNT)
XSUM2 = XSUM2 + x * x - XTIM(LPNT)^2
YSUM = YSUM + Y - YLIA(LPNT)
YSUM2 = YSUM2 + Y * Y - YLIA(LPNT)^2
XXSUM = XXSUM + x * y - XTIM(LPNT) * YLIA(LPNT)

'---------------------------------CALCULATE THE OTHER STAT Sums---------------------'
SHE(PNT) = sheval
BOL(PNT) = BOLK
BVOL(PNT) = VOLBOL
RGVOL(PNT) = VOLGAG
VSUM = VSUM + BVOL(PNT) - BVOL(LPNT)
SSUM = SSUM + SHE(PNT) - SHE(LPNT)
RGSUM = RGSUM + RGVOL(PNT) - RGVOL(LPNT)
SSUM2 = SSUM2 + SHE(PNT)^2 - SHE(LPNT)^2
BSUM = BSUM + BOL(PNT) - BOL(LPNT)
RETURN

'---------------------------------UNITS CONVERSIONS, TEMPERATURE CALCULATIONS, ETC.---------------------------------

this routine computes all of the physical values that are measured by the ADC's. It also computes the bolometer temperature from the resistance value measured by ADC(1), and computes the block temperature using the previously retrieved SHE value.

ADC(1)=LOCK IN AMPLIFIER OUTPUT (I.E. BOLOMETER VOLTAGE) Diff.In.
ADC(2)=STRAIN GAUGE HEATER VOLTAGE
ADC(3)=MAGNETIC FIELD 50 MILLI-VOLT SHUNT (S/C 6 TESLA MAGNET)
ADC(4)=ANALOG SHE OUTPUT (ON BLOCK THERMOMETER)
ADC(6)=FIXED-POINT CALIBRATION LIA VOLTAGE

VARIABLES USED/CHANGED -- VOLBOL,ADC(),OFFS(),GAIN(),OFFSET,LIGAIN,
VOLGAG,VOLSHUNT,MAGFIELD,MAGCONST,BOLRES,
SERIESR,EXVOLT,BOLK,BOLMIN(),BOLTEM,
INR,A(),BOLOMET,XEXP(),SHEVAL,GETEMP,
IFUNCT, THERM
Units conversion:

\[
\begin{align*}
\text{VOLBOL} &= \frac{\text{ADC}(1) - \text{OFFS}(1)}{(\text{gain}(1) \times 10) + \text{OFFSET}) \times \text{LIGAIN}} \\
\text{VOLGAG} &= \frac{\text{ADC}(2) - \text{OFFS}(2)}{\text{gain}(2)}: \text{ VOLTAGE ACROSS STRAIN} \\
\text{VOLSHUNT} &= \frac{\text{ADC}(3) - \text{OFFS}(3)}{\text{gain}(3)} \\
\text{MAGFIELD} &= \frac{\text{VOLSHUNT} \times \text{MAGCONST}}{\text{gain}(3)}: \text{ -- MAG FIELD IN TESLA} \\
\text{BOLRES} &= \frac{\text{VOLBOL} \times \text{SERIESR}}{(\text{EXVOLT} - \text{VOLBOL})} \\
\text{BOLK} &= \frac{\text{ABS}(\frac{\text{BOLRES}}{1000})}{\text{bolk is in k-ohms}} \\
\text{IF} \ \text{BOLRES} < \text{BOLMIN(BOLOMET) THEN} \\
\text{BOLTEMP} &= \frac{-1}{(\text{ABS}(\text{BOLK}) + .0001)} \\
\text{ELSE} \\
\text{LNR} &= \text{LOG}(\text{BOLRES}) \\
\text{BOLTEMP} &= \text{ABS}(\frac{\text{a(BOLOMET, 2)}}{(\text{LNR} - \text{a(BOLOMET, 1)})}) \\
\text{BOLTEMP} &= \text{BOLTEMP} \times (1 / \text{XEXP(BOLOMET)}) \\
\text{ENDIF} \\
\text{'-------- get the block temperature ---------------------------} \\
\text{CALL BLOCKTEMP(sheval, getemp, IFUNCT, therm) RETURN}
\end{align*}
\]

Real-time display update and slope calculator

' this routine plots the bolometer and block temperatures, prints the
' values to the display screen, checks for LIA overload, checks for
' user interrupt, checks for termination condition, and calculates
' the slope/intercept stuff to determine the termination condition
' ...

VARIABLES USED/CHANGED -- REFTIME, TIM#, BLKTEMP, GETEMP, BOLTEMP,
'icol, SHEVAL, OVLD$, BOLK, ADC(), B$, 
'FLAT, II, NPNT, DELTA, XSUM, XSUM2, YSUM, 
'YSUM2, XYSUM, SLOPE, INTERC, SIG2, SIGM, 
'ACC
'
displayupdate:

\[
\begin{align*}
\text{reftime} &= \text{tim#}: \quad \text{' set the old time} \\
\text{BLKTEMP} &= \text{getemp}
\end{align*}
\]
CALL GRAPH(reftime, BOLTEMP, icolor)
CALL GRAPH(reftime, BLKTEMP, icolor + 1)

LOCATE 21, 53: PRINT USING "###.####"; BLKTEMP
LOCATE 21, 12: PRINT USING "###.####"; sheval: : PRINT OVLDS
LOCATE 23, 54: PRINT USING "###.####"; BOLK;
PRINT "="; : PRINT USING "###.###"; BOLTEMP: : PRINT "K"
IF ABS(ADC(1)) > 9.899999 THEN BEEP: ' check if LIA is OVERLOADED

'----------------------------- look to see if we are interrupted by user -----------------------------
B$ = INKEY$: IF B$ = CHR$(13) THEN GOTO stopitnow
IF B$ <> "" THEN flat = 1

IF II >= NPNT + 1 THEN
DELTA = NPNT * XSUM2 - XSUM * XSUM
INTERC = (XSUM2 * YSUM - XSUM * XYSUM) / DELTA
SLOPE = (NPNT * XYSUM - XSUM * YSUM) / DELTA
SIG2 = (YSUM2 - INTERC * YSUM - SLOPE * XYSUM) / NPNT
SIGM = SQRT(ABS(SIG2 * NPNT / DELTA))

'----------------------------- see if slope is less than 'ACC' -----------------------------
IF (ABS(SLOPE) + SIGM) < (3 * ACC * YSUM / NPNT) THEN flat = 1
END IF

RETURN

'-------------------------------------------------------------------------------------------------------------------------------------
' MAIN DATA COLLECTION LOOP
'
'ADC    Do ADCCHAN conversions and save the voltages in ADC(i).
'Call the SHE routine, timer routine, units conversion routine,
'determine when to update PID controller, when to update the display,
'and when to collect the next data point.
'
'VARIABLES USED/CHANGED -- OLDTIME, TIM#, ZZ, ADD, ADCCHAN, ADC(), LGAIN(),
'GAIN(), OFFS(), ANSHE, SHEVAL, OVLDS$, TIMO#,
'CONTIME, TSAMP, REFTIME, REFRESH, PNTSEC
'
maindataloop:

oldtime = tim#:            ' set the old time------------------------
zz = 0:                    ' set idle loop # to zero --------------
DO
CALL GETADC(ADD, ADCCHAN, ADC(), lgain()): ' perform ADC routine
CALL SHEPCB(ADD, ADC(4), gain(4), OFFS(4), lgain(4), ANSHE, sheval, OVLD$)
CALL amdmtim(ADD, tim#, tim0#):
GOSUB unitsconversion:
IF (tim# - contime) >= tsamp THEN CALL PIDCONTROL:
IF (tim# - reftime) > REFRESH THEN GOSUB displayupdate:
zz = zz + 1:
LOOP WHILE (tim# - oldtime + .03) < 1 ! / PNTSEC
LOCATE 22, 54: PRINT USING "###.### "; tim#; : PRINT zz
RETURN

SUB amdmtim (ADDRESS, tim#, tim0#)
AMD 9513 Time READ subroutine
GLOBAL VARIABLES USED/CHANGED -- ADDRESS
TIM0#,TIM# (note double precision)
LOCAL VARIABLES USED/CHANGED -- ADD,LOW1,LOW2,HIGH1,HIGH2

Note that although the resolution of the timer is 32-bits, all of the
math calculations leading to the measurement of time MUST be 64-bit
calculations (double-precision), since it is quite common for TIM#
to exceed 1000.0000 seconds.

ADD = ADDRESS + 8:  ' timer base address
OUT ADD + 1, 163:  ' save C1 and C2 into their hold reg.
OUT ADD + 1, 17:  ' point at C1 hold register thru DPR
LOW1 = INP(ADD):
HIGH1 = INP(ADD):
OUT ADD + 1, 18:
LOW2 = INP(ADD):
HIGH2 = INP(ADD):

32-bit Time calculation

-------------------------------

TIM# = CDBL(LOW2) + CDBL(HIGH2) * 256#
TIM# = (CDBL(LOW1) + CDBL(HIGH1) * 256# + 65536# * TIM#) / 10000#:
TIM# = TIMER - TIM0#:  ' use Q-Basic's timer routine
TIM# = TIM# - TIM0#

END SUB

SUB BLOCK5K (sheval, getemp, IFUNCT) STATIC
5-k block thermometer calculations using Newton’s method of rootfinding

\[ \ln R = B_0 + B_1 T^{-0.25} + B_2 T^{-0.5} + \ldots \]

GLOBAL VARIABLES USED/CHANGED -- SHEVAL,GETEMP,IFUNCT (not changed)

LOCAL VARIABLES USED/CHANGED -- B0, B1, B2, B3, X, XRES, XPLUS1, LNR,

ITERATE, XERROR

USAGE: IFUNCT=0 MEANS CALCULATE THE TEMPERATURE FROM THE SHE
CONDUCTANCE VALUE, WHILE

IFUNCT=1 MEANS CALCULATE THE CONDUCTANCE FROM THE TEMP.

IFUNCT=2 MEANS INITIALIZE THE B? CONSTANTS

\[
\text{for } X = T^{-0.25}, \quad \frac{dT}{T} = -4 \frac{dX}{X}
\]

THE TOLERANCE OF 1E-5 SHOULD MEAN THAT WORST-CASE TEMPERATURE ERROR
divided by temperature IS 4E-5

XERROR = .00001

SELECT CASE IFUNCT

CASE 0: ' normal calculation of temp
   IF 1! / sheval < 7.4 THEN
     ' trap for out-of-calibration
     getemp = -6!
   ELSE
     LNR = -LOG(sheval / 1000!)
     x = xplus1: ' i.e. old block temp is next guess
     FOR iterate = 1 TO 15
       ' calculate f(x) and f'(x) and perform Newton's method calculation
       xplus1 = (lnr - B0 + x * x * (B2 + 2! * B3 * x))
       xplus1 = xplus1 / (B1 + x * (2! * B2 + 3! * B3 * x))
       ' note slightly different form for below exit condition
       IF ABS((x - xplus1) / x) < XERROR THEN EXIT FOR
       x = xplus1
     NEXT iterate
     getemp = 1! / xplus1 ^ 4
   END IF

CASE 1: ' calculate shevalue from temp
   x = 1! / getemp ^ .25
   LNR = B0 + x * (B1 + x * (B2 + x * B3))
   sheval = 1000! / EXP(LNR)

CASE 2: ' setup constants
   B0 = 9.652631684999999#: ' 5k block thermometer parameters
   B1 = -3.668761079#: ' APPROXIMATE FIT!!!!
   B2 = 3.905419564#
SUB BLOCKTEMP (sheval, getemp, IFUNCT, therm) STATIC

'------------------------------------------------------------------------
'   block temperature calculation
'   calculate block temperature from block thermometer
'      via Newton's method of root-finding
'
' GLOBAL VARIABLES USED/CHANGED: SHEVAL, GETEMP, IFUNCT, THERM
'
' LOCAL VARIABLES USED/CHANGED:
'
' USAGE: IFUNCT=0 MEANS CALCULATE THE TEMPERATURE FROM THE SHE
'        CONDUCTANCE VALUE, WHILE
'        IFUNCT=1 MEANS CALCULATE THE CONDUCTANCE FROM THE TEMP.
'        IFUNCT=2 MEANS INITIALIZE THE B() ARRAY
'
'        THERM=1 MEANS USE THE LO-TEMP GERMANIUM BLOCK THERMOMETER
'        THERM=2 MEANS USE THE HI-TEMP GERMANIUM BLOCK THERMOMETER
'        THERM=3 MEANS USE THE 5K Bi-Ru CHIP BLOCK THERMOMETER
'        THERM=4 MEANS USE THE GENERIC TEMPERATURE CALCULATION ROUTINE
'------------------------------------------------------------------------

SELECT CASE therm
    CASE 1
        CALL LOTEMPGE(sheval, getemp, IFUNCT)
    CASE 2
        CALL HITEMPGE(sheval, getemp, IFUNCT)
    CASE 3
        CALL BLOCK5K(sheval, getemp, IFUNCT)
    CASE 4
        CALL GENTHERM(sheval, getemp, IFUNCT)
END SELECT

END SUB

SUB DAC (ADDRESS, VOLTDAC, DACNUM)
'------------------------------------------------------------------------
DIGITAL TO ANALOG CONVERSION (DAC)

GLOBAL VARIABLES USED/CHANGED -- VOLTDAC, DACNUM, ADDRESS
LOCAL VARIABLES USED/CHANGED -- ADD, DACFACT, DACHI, DACLO

USAGE : DACNUM=0 MEANS OUTPUT 'VOLTDAC' TO DAC-0, WHILE
       DACNUM=1 MEANS OUTPUT 'VOLTDAC' TO DAC-1.
       CURRENTLY (11/08/87) DAC0 RANGE IS +/-10vdc while
       DAC1 RANGE IS +/-2.5vdc

ADD = 2 * DACNUM + ADDRESS: 'LABPAC manual for address
details
    IF DACNUM = 0 THEN DACFACT = 204.799 ELSE DACFACT = 819.199
    DACVOLT = VOLTDAC * DACFACT: 'convert voltage to binary
    DACHI = INT(DACVOLT / 256)
    DACLO = INT(DACVOLT - 256 * DACHI)
    IF DACHI < 0 THEN DACHI = DACHI + 16
    OUT ADD + 1, DACHI
    OUT ADD, DACLO

END SUB

SUB GENTHERM (sheval, getemp, IFUNCT) STATIC
    'retain values of B(), etc. between calls
    'RELATIVELY HIGH SPEED GENERAL TEMPERATURE CALCULATION ROUTINE
    '
    'IT CALCULATES THE TEMPERATURE BY NEWTON'S METHOD OF ROOT FINDING,
    'WHICH IS CONSIDERABLY FASTER THAN BISECTION, BUT NOT AS STABLE.
    'THE ODD LOOK TO THE POLYNOMIALS BELOW IS DUE TO THE USE OF
    'HORNER'S RULE FOR POLYNOMIAL EVALUATION.
    '
    'Note that on entry, SHEVAL is in MMHO's, and GETEMP is in Kelvin
    '
    'during normal operations, the number of iterations should be <= 4
    '
    GLOBAL VARIABLES USED/CHANGED -- SHEVAL, GETEMP, IFUNCT(not changed)
    '
    LOCAL VARIABLES USED/CHANGED -- B(), FX, FFX, X, XRES, XNPLUS1, XLOGRES,
    'ITERATE, IU, XERROR
    '
    USAGE: IFUNCT=0 MEANS CALCULATE THE TEMPERATURE FROM THE SHE
    'CONDUCTANCE VALUE, WHILE
    'IFUNCT=1 MEANS CALCULATE THE CONDUCTANCE FROM THE TEMP.
    'IFUNCT=2 MEANS INITIALIZE THE B() ARRAY
the constants below are for the low temperature germanium thermometer

---

SELECT CASE IFUNCT
CASE 0
  XRES = 1000 / sheval:
  XLOGRES = LOG(XRES):
  IF XRES < minres THEN
    getemp = -maxtemp:
    EXIT SUB:
  ELSEIF XRES > maxres THEN
    getemp = -mintemp:
    EXIT SUB:
  END IF
  x = xplus1:
  FOR iterate = 1 TO 10:
    GOSUB FX:
    xplus1 = -FX / FX + x
    IF ABS(xplus1 - x) < XERROR THEN EXIT FOR
  NEXT iterate
  getemp = EXP(xplus1):
  ' you may have to modify the below line for appropriate error condition
  CASE 1:
    XRES = 1000 / sheval
    XLOGRES = LOG(XRES)
    x = LOG(getemp)
  GOSUB FX
  sheval = 1000 / EXP(FX + XLOGRES)

CASE 2

' set all of the following constants to the appropriate values for the
' thermometer you wish to use. the following is for lo-temp germ.
DIM B(0 TO 20), Bi(0 TO 20)
minres = 16.01:
maxtemp = 6:
maxres = 51:
mintemp = .5:
xplus1 = 1:
XERROR = .00001:
B(0) = 3.739332503#:
B(1) = -.189034483#
B(2) = -.14310811#:
' minimum resistance allowable
' maximum temperature allowable
' maximum resistance allowable
' minimum temperature allowable
' this the initial guess
' termination condition
' initialize the constants
' STATIC stment above insures
' that the constants are
122

\[ B(3) = -0.187349917 \] retained from CALL to CALL
\[ B(4) = 0.699562002 \]
\[ B(5) = 0.46389198 \]
\[ B(6) = -1.960169743 \]'
\[ B(7) = 0.564392103 \]
\[ B(8) = 1.725619339 \]
\[ B(9) = -1.783302245 \]
\[ B(10) = 0.671232286 \]
\[ B(11) = -0.091328057 \]

' store constants for fast recall
FOR i = 0 TO 11
    Bi(i) = B(i) * CDBL(i)
NEXT i

END SELECT
EXIT SUB: ' normal EXIT

'------------function subroutine----Horner's syntax-------------
' the following is for the function
' \[ f(x) = -\log(res) + B(0) + B(1)x + B(2)x^2 + \ldots + B(11)x^{11} \]
'------------derivative subroutine----Horner's syntax-------------
' \[ f'(x) = 1*B(1) + 2*B(2)*x + \ldots + 11*B(11)*x^{10} \]
'
' further speed optimization can occur if you replace FOR-NEXT loop
'---------------------------------------------
FX: FX = B(11)
FPX: FPX = Bi(11)
FOR iu = 10 TO 1 STEP -1
    FX = B(iu) + x * FX
    FPX = Bi(iu) + x * FPX
NEXT iu
FX = -XLOGRES + B(0) + x * FX
RETURN

END SUB

SUB GETADC (ADDRESS, ADCCHAN, ADC(), lgain())
'---------------------------------------------
' 16 Bit ADC conversion routine. Get ALL 8 voltages.
',
' GLOBAL VARIABLES USED/CHANGED -- ADCCHAN,ADDRESS,ADC(),Lgain()
' LOCAL VARIABLES USED/CHANGED -- IADC,X,LOW,HIGH,T
' note that ADCCHAN should be set to 8
'---------------------------------------------
FOR IADC = 1 TO ADCCHAN
  OUT ADDRESS + 4, lgain(IADC): ' set gain and auto-increment
  '---------------ABOVE sets the amplifier gain -- 0=1, 1=10, 2=100, 3=500
  OUT ADDRESS + 6, 0: ' start conversion
  '---------------Wait until bit 7 of status byte equals 1 signaling done
  WHILE INP(ADDRESS + 4) < 128
    ' wait loop
  WEND
  LOW = INP(ADDRESS + 5): ' read in data
  HIGH = INP(ADDRESS + 6): ' reset DONE bit by reading high
  T = 256 * HIGH + LOW
  IF T > 32767 THEN T = T - 65536!
  ADC(IADC) = T / 32768. ' +/- 10vdc for 16-bit
  NEXT IADC
END SUB

SUB GETPARMS
  '-----------------------------------------------
  ' parameter initialization subroutine
  '
  ' GLOBAL VARIABLES USED/CHANGED: GAIN(), OFFS(), LGAIN(), LGAIN, OFFSET,
  '  TC, LIAFRQ, EXVOLT, SERIESR, RGAUGE,
  '  HEATRES, KGAIN, TSAMP, TINT, TDIFF, MR, THLO,
  '  THHI, TH5K, VOLTI, VOLT, VSTEPS, SETTEMP,
  '  ADCCHAN, REFRESH, ANSHE, PNTSEC, OFFSETX,
  '  OFFSETY, RANGEX, RANGEY, ACC, NPNT, FINC,
  '  STORES, THERM, PARMSFILE$
  '
  ' LOCAL VARIABLES USED/CHANGED: D$(0), G(0), O(I), IEND
  '-----------------------------------------------
  CALL PARMINIT
  DIM D$(50), G(16), O(16)
  OPEN parmsfile$ FOR INPUT AS #1

INPUT #1, G(1), G(2), G(3), G(4), G(5), G(6), G(7), G(8)
INPUT #1, LGAIN, OFFSET, TC, LIAFRQ, EXVOLT, SERIESR, RGAUGE, HEATRES
INPUT #1, O(1), O(2), O(3), O(4), O(5), O(6), O(7), O(8)
INPUT #1, kgain, tsamp, tint, tdiff, mr, THLO, THHI, TH5K
INPUT #1, VOLTI, VOLT, VSTEP, settemp, ADCCHAN, REFRESH, ANSHE, PNTSEC
INPUT #1, offsetx, rangex, offsety, rangey, ACC, NPNT, FINC, store$

  therm = THLO + 2 * THHI + 3 * TH5K
CLOSE #1

'------------------now define the name of the storage file and OPEN it------------------
store$ = store$ + "." + MID$(STR$(FINC), 2, LEN(STR$(FINC))) - 1)

OPEN store$ FOR OUTPUT AS #1
OPEN parmsfile$ FOR INPUT AS #2

'------------------read specs ------------------------------
FOR iend = 1 TO 6
   LINE INPUT #2, D$(iend)
NEXT iend
CLOSE #2

'------------------output specs to header-----------------------
FOR iend = 1 TO 6
   PRINT #1, D$(iend)
NEXT iend
PRINT #1, "DATE "; DATE$; " TIME "; TIME$;
PRINT #1, " VERSION 28 -- 3/18/89 INCLUDES SHECAL PARMS FOR 5/18/88"
PRINT #1, " TIME TEMPBOL BOLK VBOL BLKTEMP(SHE) ";
IF ANSHE = 0 THEN PRINT #1, "DIGSHE" ELSE PRINT #1, "ANSHE"

'------------------calculate the "logarithm" of the gain for the ADC routine------
FOR i = 1 TO ADCCHAN
   gain(i) = G(i)
   OFFS(i) = O(i)
   lgain(i) = INT(.5 + LOG(gain(i)) / LOG(10))
NEXT i
END SUB

SUB GRAPH (x, y, icolor)

'-------------------------------
'     real-time display routine (kill spooler and execute GRAPHICS)
'     this actually plots the data on the screen
'
' GLOBAL VARIABLES USED/CHANGED: OFFSETX,OFFSETY,RANGEX,RANGEY,
   SCRY,SCRX,X,Y,ICOLOR
'
' LOCAL VARIABLES USED/CHANGED: XP,YP,IXP,IYP
'-------------------------------

XP = (x - offsetx) / rangex: YP = (y - offsety) / rangey
IXP = (XP * scrx) \ 2 + scrx \ 2: IYP = (-YP * scry) \ 2 + scry \ 2
PSET (IXP, IYP), icolor
DEFDBL B, F, X

SUB HITEMPGE (sheval, getemp, IFUNCT) STATIC
' retain the values of B(), etc. between calls
'-------------------------------------------------------------------
' RELATIVELY HIGH SPEED TEMPERATURE CALCULATION FOR THE GR-200A-1000
' GERMANIUM THERMOMETER. IT USES THE COEFFICIENTS GIVEN IN THE
' CALIBRATION SHEET THAT CAME WITH THE THERMOMETER. IT CALCULATES THE
' TEMPERATURE BY A HEAVILY MODIFIED NEWTON'S METHOD OF ROOT FINDING.
' NOTE THAT NEWTON'S METHOD IS CONSIDERABLY FASTER THAN BISECTION,
' BUT NOT AS STABLE. THE ODD LOOK TO THE POLYNOMIALS BELOW IS DUE TO
' THE USE OF HORNER'S RULE FOR POLYNOMIAL EVALUATION.
' DO NOT USE OUTSIDE THE 1.1 TO 24 KELVIN RANGE.
'
' GLOBAL VARIABLES USED/CHANGED -- SHEVAL,GETEMP
'   IFUNCT(not changed)
'
' LOCAL VARIABLES USED/CHANGED -- B(),FX,FPX,X,XRES,XNPLUS1,XLOGRES,
'   ITERATE,I,IU,Z,BDIVX,BDIVX2,FDIVFP
'   INTEGERS#(),BI(),BIIM1(),F
' USAGE: IFUNCT=0 MEANS CALCULATE THE TEMPERATURE FROM THE SHE
'   CONDUCTANCE VALUE, WHILE
'   IFUNCT=1 MEANS CALCULATE THE CONDUCTANCE FROM THE TEMP.
'   IFUNCT=2 MEANS INITIALIZE THE B() ARRAY
'
' THE TOLERANCE OF 1E-5 SHOULD MEAN THAT WORST-CASE TEMPERATURE ERROR
' IS 1E-5 SINCE X = T
'
' note that original constants were for LOG10(RES) = ....
' so we convert to Ln(RES) = .... by multiplying all const by Ln(10)
'-------------------------------------------------------------------

SELECT CASE IFUNCT
CASE 0
    XRES = 1000! / sheval
    XLOGRES = LOG(XRES)
    IF XRES < 16.01 THEN
        getemp = 30!
        EXIT SUB
    ELSEIF XRES > 51000! THEN
        getemp = 1!
        EXIT SUB
    END IF
    IF XRES <= 729.5 THEN z = 2 ELSE z = 1

' X = 5
x = xnplus1
' this is initial guess
'
ote: x=XPPLUS1 last guess doesn't work here without modification,
' so make sure Z value is set correctly after each iteration, and make
' sure x > 0
FOR iterate = 1 TO 15:
  GOSUB HIFX:
  calculate f(x) and f'(x)
  FDIVFP = FX / FPX
  ' xnplus1 = x - FDIVFP
  ' old Newton's method
  GOTO skipmod
'
calculate highly modified Newton's Method
GOSUB HIFPPX:
  calculate f''(x)
  F = FDIVFP * FPPX / FPX
  ' this is NOT f(x)
'
X=X0 - (f'/f'') (1-sqr(1-2ff'/f''^2)) then Taylor expansion of SQR
xnplus1 = x - FDIVFP * (one# + half# * F * (one# + F))

skipmod:
  IF ABS(xnplus1 - x) < XERROR THEN EXIT FOR
  x = xnplus1
  ' assign next iteration value
  IF x < 5 THEN
    z = 1
  IF x <= 0 THEN x = 1.15: ' ensure reasonable next guess
  ELSE
    z = 2
  END IF
NEXT iterate
getemp = xnplus1
  ' assign temp on exit from FOR-NEXT
CASE 1
XRES = 1000! / sheval
  ' else calculate SHEVAL from GETEMP
XLOGRES = LOG(XRES)
x = getemp
  IF x >= 5! THEN z = 2 ELSE z = 1
GOSUB HIFX
sheval = 1000# / EXP(FX + XLOGRES)

CASE 2
DIM B(-1 TO 8), Bi(-1 TO 8, 2), BIIM(-1 TO 8, 2)
xnplus1 = 5
  ' this is T0, the initial guess
XERROR = .00001#
  ' tolerance termination condition
log10# = LOG(10#)

' constants for the function
' ln(R) = Sum-over-i-from-(-1)-to-8-of (Bi(z)*T^i)
' z=1 for T < 5K, z=2 for T >= 5K.

B(-1, 1) = 1.084791754# * log10#
B(0, 1) = 4.949677782* log10# 
B(1, 1) = -1.696463401* log10# 
B(2, 1) = .8824776532* log10# 
B(3, 1) = -.3704501521* log10# 
B(4, 1) = .1136452665* log10# 
B(5, 1) = -.0238846325#* log10# 
B(6, 1) = .00318816588* log10# 
B(7, 1) = -.0002412172369* log10# 
B(8, 1) = .000007844256543* log10# 
B(0, 2) = 4.612695665* log10# 
B(1, 2) = .8204036246999999* log10# 
B(2, 2) = .6954766986000001* log10# 
B(3, 2) = .01469722989* log10# 
B(4, 2) = -.0007335017563* log10# 
B(5, 2) = .0000152641305* log10# 
B(6, 2) = .0000001673169226* log10# 
B(7, 2) = -1.364480634D-08* log10# 
B(8, 2) = 1.813963112D-10* log10#

' store double precision products for fast recall
FOR i = 1 TO 8
   Bi(i, 1) = B(i, 1) * CDBL(i)
   Bi(i, 2) = B(i, 2) * CDBL(i)
   BIIM1(i, 1) = Bi(i, 1) * CDBL(i - 1)
   BIIM1(i, 2) = Bi(i, 2) * CDBL(i - 1)
NEXT i

half# = .5#
one# = 1#
two# = 2#

END SELECT

EXIT SUB

'------------------function subroutine--------------------------
', and
'------------------derivative subroutine------------------------
' SPEED OPTIMIZED--DO NOT ADJUST
'------------------function subroutine--------------------------

HIFX:
FX = B(6, z) + x * (B(7, z) + x * (B(8, z)))
FX = B(3, z) + x * (B(4, z) + x * (B(5, z) + x * FX))
FX = B(0, z) + x * (B(1, z) + x * (B(2, z) + x * FX))
BDIVX = B(-1, z) / x
FX = -XLOGRES + BDIVX + FX

FPX = Bi(6, z) + x * (Bi(7, z) + x * (Bi(8, z)))
FPX = Bi(3, z) + x * (Bi(4, z) + x * (Bi(5, z) + x * FPX))
\[ BDIVX2 = BDIVX / x \]
\[ FFX = -BDIVX2 + Bi(1, z) + x \times (Bi(2, z) + x \times FFX) \]

RETURN

'------------------2nd--derivative subroutine---------------------'
'
SPEED OPTIMIZED

note that this routine requires HIFX to be called first

'------------------2nd--derivative subroutine---------------------'

HIFPPX:
\[ FPPX = BIIM1(6, z) + x \times (BIIM1(7, z) + x \times (BIIM1(8, z))) \]
\[ FPPX = BIIM1(3, z) + x \times (BIIM1(4, z) + x \times (BIIM1(5, z) + x \times FPPX)) \]
\[ FPPX = two # * BDIVX2 / x + BIIM1(2, z) + x \times FPPX \]

RETURN

END SUB

SUB LOTEMPGE (sheval, getemp, IFUNCT) STATIC
' retain values of B(), etc. between calls

RELATIVELY HIGH SPEED TEMPERATURE CALCULATION FOR THE GR-200A-10
LOTEMP GERMANIUM THERMOMETER.
IT CALCULATES THE TEMPERATURE BY NEWTON'S METHOD OF ROOT FINDING.
METHOD IS CONSIDERABLY FASTER THAN BISECTION, BUT NOT AS STABLE.
THE ODD LOOK TO THE POLYNOMIALS BELOW IS DUE TO THE USE OF
HORNER'S RULE FOR POLYNOMIAL EVALUATION.
DO NOT USE OUTSIDE THE .5 TO 6 KELVIN RANGE.

Note that on entry, SHEVAL is in MMHO's, and GETEMP is in Kelvin.
during normal operations, the number of iterations should be \( \leq 4 \)
GLOBAL VARIABLES USED/CHANGED -- SHEVAL,GETEMP,IFUNCT(not changed)
LOCAL VARIABLES USED/CHANGED -- B(),FX,FFX,X,XRES,XNPLUS1,XLOGRES,
\[ \text{ITERATE, IU,} \text{ERROR} \]
USAGE: IFUNCT=0 MEANS CALCULATE THE TEMPERATURE FROM THE SHE
CONDUCTANCE VALUE, WHILE
IFUNCT=1 MEANS CALCULATE THE CONDUCTANCE FROM THE TEMP.
IFUNCT=2 MEANS INITIALIZE THE B() ARRAY

THE TOLERANCE OF 1E-5 SHOULD MEAN THAT WORST-CASE TEMPERATURE ERROR
DIVIDED BY TEMP IS $\leq 1E-5$ SINCE $T=\exp(X)$ implies that $dT/T = dX$

SELECT CASE IFUNCT
CASE 0:
  XRES = 1000! / sheval:
  XLOGRES = LOG(XRES):
  IF XRES $< \minres$ THEN
    getemp = $-\maxtemp$:
    EXIT SUB:
  ELSEIF XRES $> \maxres$ THEN
    getemp = $-\mintemp$:
    EXIT SUB:
  END IF
  $x = \text{xnplus1}$:
  FOR iterate = 1 TO 10:
    GOSUB LOFX:
    GOSUB LOFPX:
    $\text{xnplus1} = -\text{FX} / \text{FPX} + x$
    IF $AB\text{S}(\text{xnplus1} - x) < \text{XERROR}$ THEN EXIT FOR
    $x = \text{xnplus1}$:
  NEXT iterate
  getemp = $\exp(\text{xnplus1})$:
CASE 1:
  XRES = 1000! / sheval
  XLOGRES = LOG(XRES)
  $x = \log(\text{getemp})$
  GOSUB LOFX
  sheval = 1000! / $\exp(\text{FX} + \text{XLOGRES})$
CASE 2:
  DIM B(0 TO 11), Bi(0 TO 11)
  minres = 16.01:
  maxtemp = 6:
  maxres = 51!:
  mintemp = .5:
  $\text{xnplus1} = 1$:
  XERROR = .00001:
  B(0) = 3.739332503#:
  B(1) = -.189034483#:
  B(2) = -.14310811#:
  B(3) = -.187349917#:
  B(4) = .699562002#
  B(5) = .463891898#
  B(6) = -1.960169743#
  B(7) = .564392103#
  B(8) = 1.725619339#
  B(9) = -1.783302245#
  B(10) = .671232286#
B(11) = -.091328057#

' store constants for fast recall
FOR i = 0 TO 11
   Bi(i) = B(i) * CDBL(i)
NEXT i

END SELECT

EXIT SUB: ' normal EXIT

'------------------------function subroutine----Horner's syntax------------------------
' the following is for the function
' f(x)= -log(res) + B(0) + B(1)*x + B(2)*x^2 + ... + B(11)*x^11
'------------------------derivative subroutine----Horner's syntax------------------------
' f'(x) = 1*B(1) + 2*B(2)*x + ... + 11*B(11)*x^10
'
' further speed optimization can occur if you replace FOR-NEXT loop

LOFX: FX = B(11)
LOFPX: FPX = Bi(11)
   FOR iu = 10 TO 1 STEP -1
      FX = B(iu) + x * FX
      FPX = Bi(iu) + x * FPX
   NEXT iu
   FX = -XLOGRES + B(0) + x * FX
RETURN

END SUB

DEFSNG B, F, X
SUB PARMINIT

'-----------------------------------------------
' NEW METHOD OF PARAMETER INITIALIZATION -- 10/17/87
'
' GLOBAL VARIABLES USED/CHANGED: PARMFILE$, STORE$
'
' LOCAL VARIABLES USED/CHANGED: A$, B$, U$ (20, 3), V(20, 3), I, J, II, JJ, OLDRW,
   OLDCOL, OLDII, OLDJJ, OLDELN, ROWCNT, COLCNT,
   RPC, NEWLEN, BLANK$, B8$, BUILD$

INPUT "parms file (default='spec.prm')": a$
IF a$ = "" THEN a$ = "spec.prm"
parmsfile$ = a$
DIM V(20, 3), U$(20, 3)
CLS

PRINT " INITIALIZE PARAMETERS (exit by typing 'x', 'X' or <ESC>)"
PRINT
PRINT "ADC 1 GAIN= " OFFSET= VOLT INIT="
PRINT "2 = " = VOLT FINA="
PRINT "3 = " = VOLT STEP="
PRINT "4 = " = SETTEMP="
PRINT "5 = " = ADCCHAN="
PRINT "6 = " = REFRESH="
PRINT "7 = " = ANSHE="
PRINT "8 = " = POINT/SEC="
PRINT
PRINT "LIA GAIN= " PIDGAIN= OFFSET-X="
PRINT "LIA OFFSET=" PIDSAMP= RANGE-X="
PRINT "LIA TIME C= " PDIRT= OFFSET-Y="
PRINT "LIA FREQ= " PIDTDIF= RANGE-Y="
PRINT "LIA EXVOLT= " PIDMR= ACCURACY="
PRINT "LIA SERIES= " LOTEMP= N-POINTS="
PRINT "GAUGE RES= " HITEMP= FINC="
PRINT "GAUGE SERIES= " 5K-BLOK= STORE$="

OPEN a$ FOR INPUT AS #1

FOR j = 1 TO 3:
   'read/write init values of variables
   FOR i = 1 TO 16:
   'i=matrix row, j=matrix column
      IF i + j < 19 THEN
         INPUT #1, V(i, j)
         IF i = 15 AND j = 3 THEN
            V(15, 3) = V(15, 3) + 1:
            'increment 'FINC'
         END IF
         U$(i, j) = STR$(V(i, j))
      ELSE
         INPUT #1, store$
         U$(i, j) = " " + store$
      END IF
      II = i + 2 + 2 * (i \ 9): JJ = (j - 1) * 25 + 12
      LOCATE II, JJ: PRINT U$(i, j): 'ii=scrn row, jj=scrn column
   NEXT i
   NEXT j
CLOSE #1

OLDROW = 9: OLDCOL = 1: OLDII = 13: OLDJJ = 12: OLDLEN = 0
ROWCNT = 9: COLCNT = 1: RPC = 10: 'INIT CURSOR PLACEMENT--LIAGAIN

-----------------------------------
DO
RPC = ROWCNT + COLCNT
II = ROWCNT + 2 + 2 * (ROWCNT \ 9): ' account for 2 extra rows
JJ = (COLCNT - 1) * 25 + 12
LOCATE OLDII, OLDJJ: COLOR 7, 0
PRINT US$(OLDROW, OLDCOL): ' remove hi-lite from old var
LOCATE II, JJ: COLOR 16, 15
PRINT US$(ROWCNT, COLCNT): ' hi-lite new variable
NEWLEN = LEN(U$(ROWCNT, COLCNT)): BLANK$ = ""
IF NEWLEN < OLDLEN THEN BLANK$ = STRING$(OLDLEN - NEWLEN, 32)
COLOR 7, 0: PRINT BLANK$: ' tack on blank padding
OLDJJ = JJ: OLDII = II
OLDROW = ROWCNT: OLDCOL = COLCNT
OLDLEN = NEWLEN

B$ = ""
WHILE B$ = "": ' wait for a key to be pressed
   B$ = INKEY$
WEND

IF (UCASE$(B$) = "X" OR B$ = CHR$(27)) AND RPC < 19 THEN
   EXIT DO
ELSEIF LEN(B$) > 1 OR B$ = CHR$(13) THEN
   RB$ = RIGHT$(B$, 1) ' i.e. arrow key or <ret>
   BUILD$ = "": ' initialize the 'BUILD$' variable
   SELECT CASE RB$:
      CASE CHR$(77)
         COLCNT = (COLCNT MOD 3) + 1: ' -->
      CASE CHR$(75)
         COLCNT = ((COLCNT + 1) MOD 3) + 1: ' <--
      CASE CHR$(72)
         ROWCNT = ((ROWCNT + 14) MOD 16) + 1: ' /|\
      CASE CHR$(80), CHR$(13): ' <RET> = \|/
         ROWCNT = (ROWCNT MOD 16) + 1: ' \|/
   END SELECT
ELSE
   BUILD$ = BUILD$ + B$
   US$(ROWCNT, COLCNT) = " " + BUILD$
   IF RPC < 19 THEN
      V(ROWCNT, COLCNT) = VAL(BUILD$)
   ELSE
      store$ = BUILD$
   END IF
END IF
END IF
LOOP WHILE (UCASE$(B$) <> "X" AND B$ <> CHR$(27)) OR RPC = 19
OPEN a$ FOR OUTPUT AS #1
FOR j = 1 TO 3
    FOR i = 1 TO 16
        IF i + j < 19 THEN
            PRINT #1, V(i, j);
            IF i = 8 OR i = 16 THEN
                PRINT #1, ""
            ELSE
                PRINT #1, ",";
            END IF
        ELSE
            WRITE #1, store$
        END IF
    NEXT i
NEXT j
CLOSE #1
END SUB

SUB PIDCONTROL STATIC

'--------------------
' PID(limited integral action) control algorithm
'
' The sensed variable is the germanium resistor's temperature,
' and the manipulated variable is the square of the voltage
' applied to the block. Note that the germanium resistor is mounted
' in the block. The initial power level is 'MR'.
'
' GLOBAL VARIABLES: errsum, errblk, lsterr, getemp, settemp, tsamp,
' tint, tdiff, mr, kgain, tim, contime
'
' LOCAL VARIABLES: pbl, pbk
'
' contime = tim#:               ' place hooks for faster controller
' stop controller during crazy period
' IF tim# >= 15 THEN
    errblk = settemp - getemp
    errsum = errsum + errblk
    PBL = errblk + (tsamp / tint) * errsum
    PBL = PBL + (tdiff / tsamp) * (errblk - lsterr)
    PBL = kgain * PBL + mr
    IF PBL < 0 THEN PBLK = 0:   ' i.e. no imaginary voltages
    IF PBL > 100 THEN PBLK = 100: ' i.e. no voltages > 10 volts
IF (PBL > 0) AND (PBL < 100) THEN PBLK = PBL
errsum = errsum + PBLK - PBL: ' try to prevent integral windup
lerrer = errblk: ' save error for differential term
VOLTDC = SQR(PBLK): DACNUM = 0
CALL DAC(ADD, VOLTDC, DACNUM): ' perform block control (DAC0)
END IF
END SUB

SUB POWERFIT (BOLTEMP, VOLGAG, VZERO, powfuct) STATIC
'-------------------------------------------------------------
' least-squares power fitting subroutine for model
' Power=K(t^2-T0^2)=pslope*t^2+pinter
'
' GLOBAL VARIABLES USED/CHANGED: boltemp,volgag,vzero,rgauge,powfuct
'
' LOCAL VARIABLES USED/CHANGED: PXSUM,PYSUM,PXXSUM,PYYSUM,PXYSUM,U,
' X,Y,PDELTA,PSLOPE,PINTER,PSIG2,
' PSIGM,PSIGB
'
' USAGE: POWFUNCT=0 INITIALIZE STAT SUMS
' POWFUNCT=1 UPDATE THE STAT SUMS
' POWFUNCT=2 CALCULATE THE SLOPE/INTERCEPT AND UNCERTAINTIES
'
' RGAUGE IS CALCULATED VIA JING'S CALIBRATION OF STRAIN GAUGE
' FROM 2 TO 20K
' I.E. RGAUGE=R0 + R1*T + R2*T^2
' R0=RGAUGE AT OK, R1=0.1390697, AND R2=0.0017955
'-------------------------------------------------------------
SELECT CASE powfuct
CASE 0: ' entry point for sum initialization
PXSUM = 0: PYSUM = 0: PXXSUM = 0
PYYSUM = 0: PXYSUM = 0: U = 0
R1 = .1390697#
R2 = .0017955
CASE 1
'-------------------------------------------------------------
' entry point for sum update
' NOTE THAT IT IS POSSIBLE TO HAVE ELECTRICAL POWER DISSIPATION IN
' THE SAMPLE HEATER EVEN WHEN 'VOLT'=0 SINCE 'VOLGAG' MAY NOT = 0
'-------------------------------------------------------------
' note that we can calculate RGAUGE2 from the voltage across the strain
gauge via rgauge2=volgag*heater/(volt-volgag+0.000001)
' we can also approximately calculate VOLGAG from VOLT, HEATRES, RGAUGE
RGAGE2 = RGAUGE + R1 * BOLTEMP + R2 * BOLTEMP * BOLTEMP
VOLGAG = VOLT * RGAUGE2 / (RGAUGE2 + HEATRES)
Y = (VOLGAG + VZERO) ^ 2 / RGAUGE2: ' POWER DISSIPATED (DC)
x = BOLTEMP * BOLTEMP
PXSUM = PXSUM + x
PXXSUM = PXXSUM + x * x
PYXSUM = PYXSUM + x * Y
PYSUM = PYSUM + Y
PYYSUM = PYYSUM + Y * Y
U = U + 1
'
output power value and predicted gauge resistance to file
PRINT #1, " Q-dot ="; Y; " RGAUGE ="; RGAUGE2
CASE 2: ' entry point for slope/intercept calculation
PDELTA = U * PXXSUM - PXSUM * PXSUM
IF U >= 2 THEN
  IF ABS(PDELTA) < 1E-14 THEN
    PSLOPE = -1: PSIGM = -1: PINTER = -1: PSIGB = -1
  ELSE
    PINTER = (PXXSUM * PYSUM - PXSUM * PYXSUM) / PDELTA
    PSLOPE = (U * PXYSUM - PXSUM * PYSUM) / PDELTA
    PSIG2 = (PYYSUM - PINTER * PYSUM - PSLOPE * PXYSUM) / U
    PSIGM = SQR(ABS(PSIG2 * U / PDELTA))
    PSIGB = SQR(ABS(PSIG2 * PXXSUM / PDELTA))
  END IF
ENDIF
'
output results to data file
PRINT #1, " "
PRINT #1, " "
PRINT #1, " power fit: slope +/- unc, intercept +/- unc"
PRINT #1, USING "####.#####;###.##### "; PSLOPE; PSIGM; PINTER; PSIGB
PRINT #1, " "
PRINT #1, " "
END SELECT
END SUB

SUB SCREENINIT

' draw the real time display window (not the data)
'
' GLOBAL VARIABLES USED/CHANGED: OFFSETX,OFFSETY,RANGEX,RANGEY,SCRX,
SUR,SCRY,SCR
' LOCAL VARIABLES USED/CHANGED:
'
'---------------------------------------------------------------
CLS
SCREEN 9: scry = 260: ' EGA screen
SCREEN 2: scry = 150: ' CGA screen
scrx = 576: TSCRX = 150
'draw the box
LINE (0, 0)-(0, scry): LINE -(scrx, scry)
LINE -(scrx, 0): LINE -(0, 0)
LINE (0, scry \ 2)-(5, scry \ 2)
LINE (scrx \ 2, scry)-(scrx \ 2, scry - 5)
LINE (scrx - 5, scry \ 2)-(scrx, scry \ 2)
LINE (scrx \ 2, 0)-(scrx \ 2, 5)
'print the axes-scales
LOCATE 1, (scrx \ 8 + 2)
PRINT USING "###.###"; offsety + rangey
LOCATE (TSCRX * 3 \ 50 + 1), (scrx \ 8 + 2)
PRINT USING "###.###"; offsety
LOCATE (TSCRX * 3 \ 25 + 1), (scrx \ 8 + 2)
PRINT USING "###.###"; offsety - rangey
LOCATE (TSCRX * 3 \ 25 + 2), 1
PRINT USING "#####"; offsetx - rangex
LOCATE (TSCRX * 3 \ 25 + 2), (scrx \ 16 - 2)
PRINT USING "#####"; offsetx
LOCATE (TSCRX * 3 \ 25 + 2), (scrx \ 8 - 2)
PRINT USING "#####"; offsetx + rangex
END SUB

SUB SHEPCB (ADDRESS, A4, G4, O4, LG4, ANSHE, sheval, OVLDS)
DIM VSHE(4)
'---------------------------------------------------------------
'REad conductance bridge. Parallel ports are in mode 0. A as input B as
'output.
'Initialization. Address 1808+15 is 8255 control port, 1808+13 is port
'B, and 1808+12 is port A. See the data sheets on the SHE interface to
determine further data.
'
' Analog part of this routine uses ADC channel 4
'
'GLOBAL VARIABLES USED/CHANGED -- address, ADC(4), GAIN(4), LGAIN(4),
'OFFS(4), ANSHE, SHEVAL, OVLDS
LOCAL VARIABLES USED/CHANGED -- CHECK, TSHE, ISHE, ASHE, BSHE, CSHE, DSHE, VSHE(), VALUE, ANVALUE

USAGE: FOR ANSHE=0, USE THE DIGITAL INTERFACE TO GET SHEVAL.
FOR ANSHE=1, USE THE ANALOG INTERFACE TO GET SHEVAL.

Routine run.
OUT ADDRESS + 13, 10
TSHE = 4
wait for SHE to signal DONE converting
WHILE ((TSHE = 4) AND (ANSHE = 0))
    CHECK = INP(ADDRESS + 12)
    TSHE = CHECK AND 4
WEND
check for an overload condition
TSHE = CHECK AND 2
IF TSHE = 2 THEN OVLD$ = " OVERLOAD" ELSE OVLD$ = " "
FOR ISHE = 0 TO 2
    OUT ADDRESS + 13, 8 + ISHE
    VSHE(ISHE + 1) = INP(ADDRESS + 12)
NEXT ISHE
OUT ADDRESS + 13, 16
ASHE = INT(VSHE(1) / 16): ASHE = VSHE(1) - ASHE * 6
BSHE = INT(VSHE(2) / 16): BSHE = VSHE(2) - BSHE * 6
CSHE = VSHE(3) AND 1: IF VSHE(3) < 1 THEN VSHE(3) = 1
DSHE = ABS(INT(LOG(VSHE(3)) / LOG(2)))
DHE = 5
DEXP = 10 ^ (DSHE - 5)
VALUE = (CSHE * 10 ^ 4 + BSHE * 100 + ASHE) * DEXP / 10000!
SHEVOLT = (A4 - 04) / G4 + .000001

--APPLY SHE CALIBRATION CORRECTIONS OBTAINED 5/17/88 -----

SELECT CASE DSHE
CASE 1 TO 4
    TRUESHE = SHEVOLT / (.9831 + .00208 / SHEVOLT)
CASE 5
    TRUESHE = SHEVOLT / (.9831 + .00208 / SHEVOLT)
CASE 6
    TRUESHE = SHEVOLT / (.9831 + .001891 / SHEVOLT)
CASE 7
    TRUESHE = SHEVOLT / (.9807 + .001944 / SHEVOLT)
END SELECT
ANVALUE = TRUESHE * DEXP
check to see if we need to change the gain
IF G4 = 1 AND A4 <= .99 THEN
```
G4 = 10: LG4 = 1
ELSEIF G4 = 10 AND A4 >= 9.99 THEN
    G4 = 1: LG4 = 0
END IF

sheval = ABS(ANSHE * ANVALUE + (1 - ANSHE) * VALUE) + .00001

' SHEVAL = SHEVAL / 100:    ' remove from actual QP program
' SHEVAL = SHEVAL * 5:      ' these are for SIMULATOR

SUB TECINIT (ADDRESS, reftime, contime, tim0#)
'-----------------------------------------------
' Initialization routine for the SHE conductance bridge, Tecmar analog
to digital converters, and the AMD-9513 timer chip.

' GLOBAL VARIABLES USED/CHANGED: ADDRESS,REFTIME,CONTIME,TIM0#

' LOCAL VARIABLES USED/CHANGED: ADD

'-----------------------------------------------
ADDRESS = 1808:  ' TECMAR starting address = 1808

'------ INITIALIZE THE SHE ---------------
OUT ADDRESS + 15, 153:  ' 1823 is 8255 control port for SHE init

' Initialization of ADC

' Use the auto-increment feature to perform analog to digital
' conversions of channels 0 to ADCCHAN. The input range is assumed
' to be -10 to 10 volts. The board must be in the I/O mapped
' mode. The switches on the daughter board should be set to 8.
' ( #4 on, the rest off)

OUT ADDRESS + 4, 128:  ' disable autoincrement, gain=1
OUT ADDRESS + 5, 255
OUT ADDRESS + 6, 0:  ' start dummy conversion

' Wait until bit 7 of status byte equals 1 signaling done converting.
OUT ADDRESS + 4, 0:  ' enable autoincrementing
OUT ADDRESS + 5, 0:  ' starting channel=0

'-----------------------------------------------
' AMD 9513 initialization subroutine
```
This routine sets up the timer chip in the following manner:

- Counter 1 -- count source = 10khz pulse train
- Counter 2 -- count source = terminal count of counter 1
- Both counters set up as Binary counters, with net result that the timer chip counts time in 0.1ms increments with 32-bit storage
- Giving an upper time limit of 429,483.6225 seconds (approximately 5 days)

Note that although the resolution of the timer is 32-bits, all of the math calculations leading to the measurement of time MUST be 64-bit calculations (double-precision), since it is quite common for TIM# to exceed 1000.0000 seconds.

---INITIALIZATION---

```
ADD = ADDRESS + 8:  ' base address of timer
OUT ADD + 1, 255:   ' perform master reset of 9513
OUT ADD + 1, 23:    ' point to the Master-Mode register
OUT ADD, 0:         ' set MMR for BCD scaling and enable
OUT ADD, 128:       ' autoincrementing
OUT ADD + 1, 1:     ' point at C1 Mode Register thru DPR
OUT ADD, 45:        ' SET THE MODE REG C1 (SOURCE =F3)
OUT ADD, 13:        ' "
OUT ADD, 0:         ' NOW LOAD REG C1
OUT ADD, 0:         ' "
OUT ADD, 0:         ' NOW HOLD REG C1
OUT ADD, 0:         ' "
OUT ADD + 1, 233:   ' SET THE OUTPUT BIT OF REGISTER 1
OUT ADD, 8:         ' NOW MODE REG C2 (SRC=COUNTER 1 TC)
OUT ADD, 0:         ' "
OUT ADD, 0:         ' NOW LOAD REG C2
OUT ADD, 0:         ' "
OUT ADD, 0:         ' NOW HOLD REG C2
OUT ADD, 0:         ' "

---NOW READY TO START TEST SEQUENCE---

OUT ADD + 1, 232:   ' disable data pointer sequencing
OUT ADD + 1, 99:    ' Load/Arm the counters (begin count)

'tim0# = TIMER:    ' remove from regular program
tim# = 0
reftime = 0:  contime = 0: '---initialization --

END SUB
```
2. MAIN DATA ANALYZER -- SPECIFIC HEAT CALCULATOR

'--------------------------
'specific heat calculation program for the data collected by the QP
'series of data collection programs. the data sets input can either
'be the 3 variable data sets or the 6 or 7 variable data sets
'(although all must have the 6 or 7 variable header)
'
normally this program would be used on the bin-averaged data sets
'provided by the program 'AVGDATA.BAS' to help eliminate the weighting
'of the tail of the decay curve during the spline fitting part of this
'program.
'--------------------------
'DESCRIPTION OF EXTERNAL SUBROUTINES:
',
'sub boltemp() -- recalculates the 1k chip bolometer temperature for
' the data sets collected in july of 88
'
sub getpower() -- gets the power values from the file header and
' voltage steps applied to the strain gauge
'
sub graph() -- plots (x,y) data points on the monitors screen in
' the color 'icolor'
'
sub graphinit()-- initializes the screen display and prints range/
' offset information on the axes.
'
sub parminit() -- gets all of the initial parameters for the
' calculation (data file, etc.)
'
sub funcs() -- calculates the basis function values at each
' temperature used in the power fit
'
sub linfit() -- calls the svdfit routines for calculating a linear
' power fit (linear in terms of basis functions)
'
sub svd*() -- used by linfit subroutine - see "Numerical Recipes"
' for a full description
'
sub spline() -- smooths the decay data and fits a cubic spline
' polynomial to the smoothed data. this is derived
' from the IMSL routine ICSSCU
'--------------------------
'MODIFICATION HISTORY OF CPT.BAS
4/08/89 -- STRAIN GAUGE CALIBRATION INSTALLED

the strain gauge calibration data measured by Jing has been installed in the 'getpower' subroutine

DEFSNG A-H, O-Z

COMMON SHARED offsetx, rangey, offsety, rangey, SCRX, scry, tscry
COMMON SHARED b0, b1, b2, b3, nmax, mmax, tol, ic, ega$
COMMON SHARED nmes$, bastemp, maxexc, mass, recalc$, stront, order
COMMON SHARED RGAUGE, seriesr, powfact, nlines, itype
COMMON SHARED formula$$, molwgt$$, basemod$$, begbase$$, endbase

DECLARE SUB bolttemp (res, TEMP)
DECLARE SUB getpower (xx(), yy(), L)
DECLARE SUB graph (X, y, icolor)
DECLARE SUB graphinit ()
DECLARE SUB parminit ()
DECLARE SUB funcs (X, AP(), NP)
DECLARE SUB LINFIT (X(), y(), sg(), N, A(), M, NP, MF, cm(), CHSQ)
DECLARE SUB SPLINE (X(), f(), DF(), N, sm, y(), C(), ic, WK(), er)
DECLARE SUB SVBKS B (U(), W(), V(), M, N, MP, NP, b(), X())
DECLARE SUB SVDFIT (X(), y(), s(), N, A(), M, U(), V(), W(), MP, NP, Q)
DECLARE SUB SVDCMP (A(), M, N, MP, WP, W(), V())
DECLARE SUB SVDVAR (V(), MA, NP, W(), CVM(), ncvm)

' set the maximum number of data points, and the maximum order of polynomial fit for the power fitting subroutine

  nmax = 1000: mmax = 6: tol = .000001: ic = nmax - 1

DIM X(nmax), y(nmax), sig(nmax), A(mmax)
DIM CVM(mmax, mmax)
DIM xx(150), yy(150), molwgt(5)

DIM f(nmax), DF(nmax), C(ic, 3), WK(7, nmax)
DIM Y1(nmax), Y2(nmax), Y3(nmax), Y4(nmax), Y5(nmax)

ega$ = "Y"

' reconfigured system implies the bolometer calibration is no longer valid.

'
icolor = 2

b0 = 6.759626#: ' constants for recalibration, if necessary
b1 = -.05724#: ' lnR = B0 + B1*T^(-.25) + B2*T^(-.5) NOT 1k-params!!
b2 = .951276#:'
b3 = 0

RGAUGE = 109.1: ' STRAIN-GAUGE RESISTANCE AT 4.2K
seriesr = 5740: ' SERIES RESISTANCE WITH STRAIN GAUGE
OLDLEN = 11: L = 0

molwgt(1) = 734.56: ' mole weight for GdBaSr(.00)CuO
molwgt(2) = 722.13: ' mole weight for GdBaSr(.25)CuO
molwgt(3) = 709.7: ' mole weight for GdBaSr(.50)CuO
molwgt(4) = 1
molwgt(5) = 1

' get the file name, base temp, sample mass, etc., then read the power
' level information from the data file and calculate the power-losses

DO
  CALL parminit
  CLS
  L = 0
  CALL getpower(xx(), yy(), L)

  PRINT "CALCULATE power again?(y/n)";

  ' -- look for any keystroke-----------------------------------------------
  DO
    A$ = INKEYS
    LOOP WHILE A$ = ""

  LOOP WHILE UCASE$(A$) = "Y"

  ' initialize the graphics display and plot the power-loss cal points

  offsetx = (xx(1) + xx(L)) / 2!
  rangex = (xx(L) - xx(1)) / 1.9
  rangey = yy(L) / 1.9
  offsety = rangey

  DO
    CALL graphinit: ' initialize graphics
FOR i = 1 TO L
    sig(i) = 1: ' set the standard dev
    CALL graph(xx(i), yy(i), icolor): ' plot the power data
NEXT i

LOCATE 20, 60: PRINT "plot power again?";
' -- look for any keystroke-----------------------------
DO
    A$ = INKEY$ 
LOOP WHILE A$ = "" 

LOOP WHILE UCASE$(A$) = "Y"

' now calculate the power fits
' np >= ndata, np >= ma, ma = # of coefficients,
' ndata = # of data points, and here, ndata=L, ma=ORDER
' LINFIT performs the polynomial fit to data (usually power fits) via
' Singular Value Decomposition ....

NP = order: MP = L
CALL LINFIT(xx(), yy(), sig(), L, A(), order, NP, MP, CVM(), chisq)

LOCATE 20, 1: PRINT "A1 ="; A(1); " +/- "; SQR(CVM(1, 1)) 
LOCATE 21, 1: PRINT "A2 ="; A(2); " +/- "; SQR(CVM(2, 2)) 
LOCATE 22, 1: PRINT "A3 ="; A(3); " +/- "; SQR(CVM(3, 3))

MAXX = offsetx + rangex: MINN = offsetx - rangex

FOR z = CSNG(MINN) TO CSNG(MAXX) STEP CSNG((MAXX - MINN) / 100#)
    X = z

    y = A(order): ' evaluate using Horner's rule
    FOR N = order - 1 TO 2 STEP -1
        y = A(N) + X * y
    NEXT N
    y = A(1) + X * X * y

    CALL graph(X, y, icolor + 1)
NEXT z

' done with power fit, now try to calculate the heat capacity
' via the decay curve

OPEN nme$ FOR INPUT AS #1
'---- get past the power calibration points

FOR i = 1 TO nlines + 6
    LINE INPUT #1, QQ$
NEXT i

INDEX = 0

'---------begin reading the decay curve-----------------------------

DO
    IF itype < 6 THEN
        INPUT #1, time, TEMP, blktemp
        ON ERROR GOTO donereadingdata
    ELSEIF itype = 6 THEN
        INPUT #1, time, TEMP, res, MAG50MV, blktemp, SHECON
        ON ERROR GOTO donereadingdata
    ELSEIF itype > 6 THEN
        INPUT #1, time, TEMP, res, MAG50MV, FIXPT, blktemp, SHECON
        ON ERROR GOTO donereadingdata
    END IF

' automatically subtract off starting time
    IF INDEX = 0 THEN toff = time
    time = time - toff

' IF ABS(time - OLDTIME) > 10 THEN icolor = icolor + 1
    OLDTIME = time
    IF recalc$ = "Y" THEN CALL boltemp(res, TEMP)

' IF ((temp - bastemp) < .05) AND (INDEX > 25) THEN EXIT DO

INDEX = INDEX + 1

' pick a value for the exponential decay model of the base temperature
    test = (TEMP - bastemp) / (xx(L) - bastemp)
    IF test <= .4 AND test > .35 THEN basetau = time

    X(INDEX) = time
    f(INDEX) = TEMP
    IF (INDEX MOD 10) = 0 THEN
        LOCATE 22, 20: PRINT "INDEX="; INDEX
    END IF
LOOP WHILE INKEY$ = ""

donereadingdata:

CLOSE #1
IF INDEX < 3 THEN GOTO wearedone
'--ASSIGN THE DEFAULT WEIGHT FOR SPLINE OF 1

weight = 1!
FOR i = 1 TO INDEX
'
correct for base temperature shift
   IF basemod$ = "y" THEN
      tempcorr = (begbase - endbase) / (xx(L) - bastemp)
      tempcorr = tempcorr * (TEMP - bastemp)
      f(i) = f(i) - tempcorr
   END IF
   DF(i) = weight:  ' ASSIGN WEIGHT
NEXT i
'
' use old temperature range from power fit for offsetx, rangex
offsetx = 10: rangex = 10

CALL graphinit:  ' INITIALIZE DISPLAY

splineloop:

   LOCATE 6, 3: INPUT "smooth (default=1) ="; A$
   LOCATE 7, 3: INPUT "storage(default='E:temp.spc')"; storage$
   IF storage$ = "" THEN storage$ = "E:temp.spc"
   IF A$ <> " " THEN smooth = VAL(A$) ELSE smooth = 1
   OPEN storage$ FOR OUTPUT AS #2

   LOCATE 3, 5: PRINT "FILE = "; nmes$
   LOCATE 4, 5: PRINT "DATE = "; DATE$
   LOCATE 5, 5: PRINT "TIME = "; TIME$

   CALL SPLINE(X(), f(), DF(), INDEX, smooth, Y1(), C(), ic, WK(), ierl)

   IF ierl <> 0 THEN
      PRINT " error spline # "; ierl
      GOTO wearedone
   END IF

' now calculate the specific heats

' use the predicted value of temperature and derivative of temperature
' provided via the SPLINE subroutine

   FOR nn = 2 TO INDEX
      ctemp = Y1(nn)
      tdot = C(nn, 1)
   NEXT nn

' calculate the power from the power fit above
   power = A(order):  ' evaluate using Horner's rule
   FOR N = order - 1 TO 2 STEP -1
power = A(N) + ctemp * power
NEXT N
power = A(1) + power * ctemp * ctemp

'-------- joule/mole-K -------------------------------
IF ABS(tdot) < 1E-08 THEN EXIT FOR
Cp = (-power / tdot) * molwgt(formula%) / mass
cdivt = Cp / ctemp
time = X(nn)
PRINT #2, USING "#.###^##" ; ctemp; cdivt; power; time
X = ctemp
Y = cdivt
IF (nn MOD 10) = 0 THEN
b$ = INKEY$
IF b$ <> "" THEN EXIT FOR
LOCATE 21, 25: PRINT "TEM="; PRINT USING "###.### "; X
LOCATE 22, 19: PRINT "Cp / temp =";
PRINT USING "#.###^##"; Y
LOCATE 22, 1: PRINT "INDEX="; nn
END IF
CALL graph(CDBL(ctemp), CDBL(cdivt), icolor)
NEXT nn
CLOSE #2

LOCATE 22, 60: PRINT "spline again"; : INPUT dummy$
IF UCASE$(dummy$) = "Y" THEN GOTO splineloop

wearedone:

SCREEN 0
STOP
END

SUB boltemp (res, TEMP)
'-----------------------------------------------
' this subroutine was needed for all data sets taken during July 88,
' which suffered from an error in the original 1k-ohm bolometer
' calibration. This bolometer was recalibrated in September 88.
'
' GLOBAL VARIABLES USED/CHANGED: B0,B1,B2,RES,TEMP
'
' LOCAL VARIABLES USED/CHANGED: TT
$$TT = -b1 + SQR(b1^2 - 4*b2*(b0 - LOG(1000*b1/res)))$$
$$TT = TT / (2*b2)$$
$$TEMP = 1/TT^4$$

END SUB

SUB funcs(X, AP(), NP)
' the basis functions calculated below are X^0, X^1, X^2, ....
'
' note that there is no linear term since there is no physical reason
' for a linear term. the power fitting function is essentially
' 
' $$P = A1 + A2*T^2 + A3*T^3 + ...$$
'
' GLOBAL VARIABLES USED/CHANGED: X, AP(), NP
'
' LOCAL VARIABLES USED/CHANGED: X, AP(), NP, J

AP(1) = 1!
AP(2) = X * X
FOR J = 3 TO NP
    AP(J) = AP(J - 1) * X
NEXT J
END SUB

SUB getpower(temper(), powers(), L)
' this subroutine extracts the heater powers supplied to the sample
' heater. these values are calculated by reading the header at the
' beginning of the data sets. if a recalculation of temperature is
' necessary, it is done in this routine. if a correction for base
' temperature drift is necessary, it is performed in this routine.
'
' GLOBAL VARIABLES USED/CHANGED: NME$, TEMPER(), POWERS(), L, RGAUGE, ITYPE,
' RECALC$, POWFACT
PRINT "data file="; nmex$
ndum = 8: ' # of dummy lines at beginning of file

OPEN nmex$ FOR INPUT AS #1
INPUT #1, U, GAIN2, U, U, U, U, U
INPUT #1, U, VEX0, U, U, U, U, U, U

FOR i = 3 TO ndum
   LINE INPUT #1, QQ$
NEXT i

nlines = ndum
LENCHAR = 12: ITEMP = 2: IRES = 3
PRINT "temperature deltemp res power dcexc"

' ------- start reading lines from file for power fit----------
WHILE NOT EOF(1)
   LINE INPUT #1, QQ$
   ON EOF(1) GOTO probablydone
   newlen = LEN(QQ$)
   IF newlen < 10 THEN GOTO probablydone
   ' extract the bolometer heater voltage
   FOR i = 26 TO 27
      IF MID$(QQ$, i, 1) = "=" THEN EQ = i
   NEXT i
   vex = VAL(MID$(QQ$, EQ + 1, newlen - EQ))
   IF iype = 6 THEN
      INPUT #1, time, TEMP, res, MAG50MV, blktemp, SHECON
   ELSE
      INPUT #1, time, TEMP, res, MAG50MV, FIXPT, blktemp, SHECON
   END IF

OLDTIME = time
LINE INPUT #1, QQ$
LINE INPUT #1, QQ$: REM read dummy lines
nlines = nlines + 4
IF recalc$ = "Y" THEN .
CALL boltemp(res, TEMP): REM re-calculate bol temp
END IF

'-------
dcvolt=vex*res/(res+seriesr)-----------------------------
IF vex = 0 THEN bastemp = TEMP

' we still need to read the power data even if vex > maxexc because we
' need to know the number of dummy lines to read
IF vex <= maxexc THEN
  DELTEMP = TEMP - bastemp
  ' install the calibration for the strain gauge for 0-20K
  RGAUGE2 = RGAUGE + .1390697* TEMP + .0017955 * TEMP * TEMP
  powfact = RGAUGE2 / (RGAUGE2 + seriesr) ^ 2

  power = vex * vex * powfact
  POWER0 = ABS(VEX0 / GAIN2) ^ 2 / RGAUGE2
  IF vex = 0 THEN power = POWER0
  L = L + 1: temper(L) = TEMP: powers(L) = power
  PRINT USING "###.####"; TEMP; DELTEMP;
  PRINT USING "###.####"; res;
  PRINT USING "####.####"; power; vex
END IF
WEND

probablydone:

CLOSE #1

IF basemod$ = "Y" THEN
  FOR i = 1 TO L
    tempcorr = (begbase - endbase) / (temper(L) - temper(1))
    tempcorr = (temper(i) - temper(1)) * tempcorr
    temper(i) = temper(i) - tempcorr
  NEXT i
END IF
END SUB

SUB graph (X, y, icolor)

' real-time display routine (kill spooler and execute GRAPHICS)
' this subroutine plots the data points (X,Y) onto the display screen
'
it is assumed that the range and offset variables are passed to this
'subroutine via the COMMON shared block in the main calling routine
'
GLOBAL VARIABLES USED/CHANGED: RANGEX, RANGEXY, OFFSETX, OFFSETY, SCRX,
                        SCRY, ICOLOR, X, Y
'
LOCAL VARIABLES USED/CHANGED: X, Y, IP, XP, YP

XP = (X - offsetx) / rangex
YP = (Y - offsety) / rangey
IP = (XP * SCRX) \ 2 + SCRX \ 2
IY = (-YP * SCRY) \ 2 + SCRY \ 2
PSET (IP, IY), icolor

END SUB

SUB graphinit

' graphics initialization routine
'
GLOBAL VARIABLES USED/CHANGED: RANGEX, RANGEXY, OFFSETX, OFFSETY, SCRX,
                        SCRY, TSCRX, EGS
'
LOCAL VARIABLES USED/CHANGED: V$

PRINT "real-time display attributes:"
PRINT "X-center, X-range, Y-center, Y-range"
PRINT "(defaults are ="; offsetx, rangex, offsety, rangey; ")"

INPUT "offsetx ="; V$: IF V$ <> "" THEN offsetx = VAL(V$)
INPUT "rangex ="; V$: IF V$ <> "" THEN rangex = VAL(V$)
INPUT "offsety ="; V$: IF V$ <> "" THEN offsety = VAL(V$)
INPUT "rangey ="; V$: IF V$ <> "" THEN rangey = VAL(V$)

REM draw the real-time display box
CLS
IF UCASE$(ega$) = "Y" THEN
  SCRY = 260
  SCREEN 9
ELSE
   scry = 150
   SCREEN 2
END IF

SCRX = 576: tscry = 150

LINE (0, 0)-(0, scry)
LINE -(SCRX, scry)
LINE -(SCRX, 0)
LINE -(0, 0)
LINE (0, scry \ 2)-(5, scry \ 2)
LINE (SCRX \ 2, scry)-(SCRX \ 2, scry - 5)
LINE (SCRX - 5, scry \ 2)-(SCRX, scry \ 2)
LINE (SCRX \ 2, 0)-(SCRX \ 2, 5)
LOCATE 1, (SCRX \ 8 + 2)
PRINT USING "#.##^####"; offsety + rangey
LOCATE (tscry * 3 \ 50 + 1), (SCRX \ 8 + 2)
PRINT USING "#.##^####"; offsety
LOCATE (tscry * 3 \ 25 + 1), (SCRX \ 8 + 2)
PRINT USING "#.##^####"; offset - rangey
LOCATE (tscry * 3 \ 25 + 2), 1
PRINT USING "###.###"; offsetx - rangex
LOCATE (tscry * 3 \ 25 + 2), (SCRX \ 16 - 2)
PRINT USING "###.###"; offsetx
LOCATE (tscry * 3 \ 25 + 2), (SCRX \ 8 - 2)
PRINT USING "###.###"; offsetx + rangex

END SUB

SUB LINFIT (X(), Y(), SIG(), NDATA, A(), MA, NP, MP, CVM(), CHISQ)
'--------------------------------------------------------------
' set aside some workspace for the singular value decomposition
',
' GLOBAL VARIABLES USED/CHANGED: X(),Y(),SIG(),A(),CVM(),NDATA,MA,MP,
',
',           CVM(),CHISQ
',
' LOCAL VARIABLES USED/CHANGED: U(),V(),W(),NN
'--------------------------------------------------------------
   DIM U(nmax, mmax), V(mmax, mmax), W(mmax)
',
' mp >= ndata, np >= ma, ma = # of coefficients,
',
' ndata = # of data points
nn = NDATA
CALL SVDFIT(X(), y(), sig(), nn, A(), MA, U(), V(), W(), MP, NP, chisq)
NDATA = nn
'
calculate the uncertainties in the power fit parameters
    CALL SVDDVAR(V(), MA, NF, W(), CVM(), MA)

END SUB

SUB parminit
'----------------------------------------------------------------------
' get all of the parameters for the specific heat calculation
',
' GLOBAL VARIABLES USED/CHANGED: NME$, ITYPE, BASTEMP, MAXEXC, MASS, STRONT,
', ORDER, BASEMOD$, BEGBASE, ENDBASE,
', RECALC$
',
' LOCAL VARIABLES USED/CHANGED: A$
'----------------------------------------------------------------------
OPEN "gdcp.prm" FOR INPUT AS #3
INPUT #3, nme$
INPUT #3, itype
INPUT #3, bastemp, maxexc, mass, stront, order
INPUT #3, baseMOD$
INPUT #3, begbase, endbase
INPUT #3, recalC$
CLOSE #3

PRINT "data file name (default="; nme$; ")"; : INPUT A$
IF A$ <> "" THEN nme$ = A$
PRINT "data file type (3, 6, or 7 variables. default="; itype; ")";
INPUT A$: IF A$ <> "" THEN itype = VAL(A$)
PRINT "base temperature (default="; bastemp; ")"; : INPUT A$
IF A$ <> "" THEN bastemp = VAL(A$)
PRINT "max dc excitation voltage to consider (default=";
PRINT maxexc; ")";
INPUT A$: IF A$ <> "" THEN maxexc = VAL(A$)
PRINT "sample mass (default="; mass; ")"; : INPUT A$
IF A$ <> "" THEN mass = VAL(A$)
'
find out which formula we are using for the molecular weight
' calculation
    PRINT "Sr amount, e.g. 0, .25, .5 (default="; stront; ")";
INPUT A$: IF A$ <> "" THEN stront = VAL(A$)

SELECT CASE stront
    CASE .24 TO .26
        formula% = 2
    CASE .45 TO .55
        formula% = 3
    CASE ELSE
        formula% = 1
END SELECT

PRINT "reacalculate bolometer temperature (default="; recalc$; ")"
INPUT A$: IF A$ <> "" THEN recalc$ = A$
IF recalc$ = "y" THEN recalc$ = "Y"
PRINT "order of polynomial power fit (max=6,default="; order; ")"
INPUT A$: IF A$ <> "" THEN order = VAL(A$)

' find out if we are adjusting the base temperature via an exponential decay model
PRINT "base temperature correction (default="; basemod$; ")"
INPUT A$: IF A$ <> "" THEN basemod$ = A$
IF basemod$ = "y" OR basemod$ = "Y" THEN
    basemod$ = "Y"
PRINT "beginning base temp (default="; begbase; ")"
INPUT A$: IF A$ <> "" THEN begbase = VAL(A$)
PRINT "ending base temp (default="; endbase; ")"
INPUT A$: IF A$ <> "" THEN endbase = VAL(A$)
END IF

' write out the parameters
OPEN "gdcpprm" FOR OUTPUT AS #3
PRINT #3, nme$
PRINT #3, itype
PRINT #3, bastemp, maxexc, mass, stront, order
PRINT #3, basemod$
PRINT #3, begbase, endbase
PRINT #3, recalc$
CLOSE #3

END SUB

SUB SPLINE (X(), f(), DF(), nx, sm, y(), C(), ic, WK(), ier)
'------------------------------------------------------------
SMOOTHING CUBIC SPLINE ROUTINE

X()  -- vector of length NX containing the X coordinates of the NX data points (X(i),F(i)). Note that X(i) must be ordered such that X(i) > X(i+1)    INPUT

F()  -- vector of length NX containing the Y coordinates of the NX data points (X(i),F(i))    INPUT

DF() -- vector of length NX containing the relative weights of data point 'i'    INPUT

NX  -- number of data points; must be >= 2    INPUT

SM  -- non-negative number that controls the extent of smoothing. the spline function S() is determined such that the sum from 1 to NX of
      SUM-OVER-i ((S(X(i))-F(i))/DF(i))^2 <= SM
      where equality holds unless S describes a straight line    INPUT

Y,C  -- spline coefficients. Y() is a vector of length NX. C() is an NX-1 by 3 matrix. The value of the spline approximation at T is
      S(T)=((C(i,3)*D+C(i,2))*D+C(i,1))*D+Y(i)
      i.e.
      S(T)=C(i,3)*D^3+C(i,2)*D^2+C(i,1)*D+Y(i)
      and
      S'(T)=3*C(i,3)*D^2+2*C(i,2)*D+C(i,1)
      where X(i) <= T < X(i+1) and D=T-X(i)    OUTPUT

IC  -- Row dimension of matrix C exactly as specified in the DIM statement in the calling program    INPUT

WK() -- work area vector of length 7*NX+14    I/O

IER  -- error parameter: IER=129 is IC < NX-1 OUTPUT
       IER=130 is NX < 2
       IER=131 is input X(i) not ordered

GLOBAL VARIABLES USED/CHANGED: X(),F(),DF(),NX,SM,Y(),C(),IC,WK(),IER

LOCAL VARIABLES USED/CHANGED: ZERO,ONE,ONED3,TWOD3,M2,NP1,F,F2,FF,
                                E,F,G,H,I,J,ONEDH,ZZZ,NP3,HMG

' subroutine derived from IMSL routine 'icsscu'
'-------------------------
'--USEFUL CONSTANTS FOR FAST ACCESS--
-------------------------

twod3 = .6666667
oned3 = .3333333
zero = 0!
one = 1!
ier = 0

IF ic < (nx - 1) THEN
  ' row dimension of C needs to be >= NX
  ier = 129
  GOTO 9005
ELSE IF nx < 2 THEN
  ' need more than 2 data points
  ier = 130
  GOTO 9005
END IF

M2 = nx + 2
NPl = nx + 1
WK(1, 1) = zero
WK(1, 2) = zero
WK(2, NPl) = zero
WK(3, M2) = zero
WK(3, NPl) = zero
WK(6, 1) = zero
WK(6, 2) = zero
WK(6, M2) = zero
WK(6, NPl) = zero
P = zero

H = X(2) - X(1)

' H < 0 means X(i) is not ordered
IF H <= zero THEN
  ier = 131
  GOTO 9005
END IF

F2 = -sm
FF = (f(2) - f(1)) / H
IF nx < 3 THEN GOTO 2030

FOR i = 3 TO nx
  G = H:
  H = X(i) - X(i - 1):

  ' H < 0 means X(i) is not ordered
  IF H <= zero THEN
    ier = 131
    GOTO 9005
  END IF
ONEDH = one / H
E = FF:
OLD SLOPE FF
FF = (f(i) - f(i - 1)) * ONEDH
y(i) = FF - E
WK(4, i) = (G + H) * twod3
WK(5, i) = H * oned3
WK(3, i) = DF(i - 2) / G
WK(1, i) = DF(i) * ONEDH
WK(2, i) = -DF(i - 1) / G - DF(i - 1) * ONEDH
NEXT i

FOR i = 3 TO nx
C(i - 1, 1) = WK(1, i) * WK(1, i) + WK(2, i) * WK(2, i)
C(i - 1, 1) = C(i - 1, 1) + WK(3, i) * WK(3, i)
C(i - 1, 2) = WK(1, i) * WK(2, i + 1) + WK(2, i) * WK(3, i + 1)
C(i - 1, 3) = WK(1, i) * WK(3, i + 2)
NEXT i

2015 IF nx < 3 THEN GOTO 2030

FOR i = 3 TO nx
WK(2, i - 1) = FF * WK(1, i - 1)
WK(3, i - 2) = G * WK(1, i - 2)
zzz = WK(4, i) - FF * WK(2, i - 1) - G * WK(3, i - 2)
WK(1, i) = one / (P * C(i - 1, 1) + zzz)
zzz = y(i) - WK(2, i - 1) * WK(6, i - 1)
WK(6, i) = zzz - WK(3, i - 2) * WK(6, i - 2)
FF = P * C(i - 1, 2) + WK(5, i) - H * WK(2, i - 1)
G = H
H = C(i - 1, 3) * P
NEXT i

NP3 = nx + 3

FOR i = 3 TO nx
J = NP3 - i
zzz = WK(1, J) * WK(6, J) - WK(2, J) * WK(6, J + 1)
WK(6, J) = zzz - WK(3, J) * WK(6, J + 2)
NEXT i

2030 E = zero
H = zero

FOR i = 2 TO nx
G = H
H = (WK(6, i + 1) - WK(6, i)) / (X(i) - X(i - 1))
HMG = H - G
WK(7, i) = HMG * DF(i - 1) * DF(i - 1)
E = E + WK(7, i) * HMG
NEXT i

G = -H * DF(nx) * DF(nx)
WK(7, NP1) = G
E = E - G * H
G = F2
F2 = E * P * P
IF (F2 >= sm) OR (F2 <= G) THEN GOTO 2050
FF = zero
H = (WK(7, 3) - WK(7, 2)) / (X(2) - X(1))
IF NOT (nx < 3) THEN

FOR i = 3 TO nx
G = H
H = (WK(7, i + 1) - WK(7, i)) / (X(i) - X(i - 1))
zzz = WK(3, i - 2) * WK(1, i - 2)
G = H - G - WK(2, i - 1) * WK(1, i - 1) - zzz
FF = FF + G * WK(1, i) * G
WK(1, i) = G
NEXT i

END IF

H = E - P * FF
IF NOT (H <= zero) THEN
P = P + (sm - F2) / ((SQR(sm / E) + P) * H)
GOTO 2015
END IF

2050 NP1 = nx - 1

FOR i = 1 TO NP1
y(i) = f(i) - P * WK(7, i + 1)
C(i, 2) = WK(6, i + 1)
WK(1, i) = y(i)
NEXT i

WK(1, nx) = f(nx) - P * WK(7, nx + 1)
y(nx) = WK(1, nx)

FOR i = 2 TO nx
H = X(i) - X(i - 1)
C(i - 1, 3) = (WK(6, i + 1) - C(i - 1, 2)) / (H + H + H)
zzz = H - (H * C(i - 1, 3) + C(i - 1, 2)) * H
C(i - 1, 1) = (WK(1, i) - y(i - 1)) / zzz
SUB SVBKSB (U(), W(), V(), M, N, MP, NP, b(), X())

' From "Numerical Recipes", converted to BASIC by "FORT2BAS.BAS"
'
' solves A.x = b for a vector X, where A is specified by the arrays U,
' V, W, as returned by SVDCMP. M and N are the logical dimensions of A,
' and will be equal for square matrices. MP and NP are the physical
' dimensions of A. B is the input right hand side. X is the output
' solution vector. No input quantities are destroyed, so the routine
' may be called sequentially with different B's. M must be >= N;
' see SVDCMP.
'
DIM tmp(nmax)
'
calculate U transpose * B
FOR J = 1 TO N
    s = 0!
    ' non-zero result only if W-sub-j is nonzero
    IF (W(J) <> 0!) THEN
        FOR i = 1 TO M
            s = s + U(i, J) * b(i)
        NEXT i
    s = s / W(J)
    tmp(J) = s
NEXT J
'
matrix multiply by V to get answer
FOR J = 1 TO N
    s = 0!
    FOR JJ = 1 TO N
        s = s + V(J, JJ) * tmp(JJ)
    NEXT JJ
    X(J) = s
NEXT J
END SUB
SUB SVDCMP (A(), M, N, MP, NP, W(), V())

' From "Numerical Recipes", converted to BASIC by "FORT2BAS.BAS"

' Given a matrix A, with logical dimensions M by N and physical
dimensions MP by NP, this routine computes its singular value
decomposition, A = U.'W'.V transpose. The matrix U replaces A on
output. The diagonal matrix of singular values 'W' is output as a
vector W. The matrix V (not the transpose of V) is output as V.
M must be >= N; if it is smaller, then A should be filled up to
square with zero rows.

DIM RV1(nmax)

' Householder reduction to bi-diagonal form
IF (M < N) THEN PRINT "you must augment A with extra zero rows"
G = 0!
SCALE = 0!
ANORM = 0!

FOR i = 1 TO N
L = i + 1
RV1(i) = SCALE * G
G = 0!
s = 0!
SCALE = 0!
IF (i <= M) THEN
FOR K = i TO M
SCALE = SCALE + ABS(A(K, i))
NEXT K
IF (SCALE <> 0!) THEN
FOR K = i TO M
A(K, i) = A(K, i) / SCALE
s = s + A(K, i) * A(K, i)
NEXT K
f = A(i, i)
G = -ABS(SQR(s)) * SGN(f)
H = f * G - s
A(i, i) = f - G
IF (i <> N) THEN
FOR J = L TO N
s = 0!
FOR K = i TO M
\[ s = s + A(K, i) \times A(K, J) \]

NEXT K

\[ f = s / H \]

FOR K = i TO M

\[ A(K, J) = A(K, J) + f \times A(K, i) \]

NEXT K

NEXT J

END IF

FOR K = i TO M

\[ A(K, i) = SCALE \times A(K, i) \]

NEXT K

END IF

END IF

W(i) = SCALE \times G

G = 0!

s = 0!

SCALE = 0!

IF ((i <= M) AND (i <> N)) THEN

FOR K = L TO N

\[ SCALE = SCALE + \text{ABS}(A(i, K)) \]

NEXT K

IF (SCALE <> 0!) THEN

FOR K = L TO N

\[ A(i, K) = A(i, K) \div SCALE \]

\[ s = s + A(i, K) \times A(i, K) \]

NEXT K

f = A(i, L)

G = -\text{ABS(\text{SQR(s)})} \times \text{SGN(f)}

H = f \times G - s

\[ A(i, L) = f - G \]

FOR K = L TO N

\[ \text{RV1}(K) = A(i, K) / H \]

NEXT K

IF (i <> M) THEN

FOR J = L TO M

s = 0!

FOR K = L TO N

\[ s = s + A(J, K) \times A(i, K) \]

NEXT K

FOR K = L TO N

\[ A(J, K) = A(J, K) + s \times \text{RV1}(K) \]

NEXT K

NEXT J

END IF

FOR K = L TO N

\[ A(i, K) = SCALE \times A(i, K) \]

NEXT K


END IF

END IF

uuuuu = ABS(W(i)) + ABS(RV1(i))
IF ANORM < uuuuu THEN ANORM = uuuuu

NEXT i

' accumulation of right-hand transformations.
FOR i = N TO 1 STEP -1
  IF (i < N) THEN
    IF (G <> 0!) THEN
      ' double division to avoid possible underflow
      FOR J = L TO N
        V(J, i) = (A(i, J) / A(i, L)) / G
      NEXT J
      FOR J = L TO N
        s = 0!
        FOR K = L TO N
          s = s + A(i, K) * V(K, J)
        NEXT K
        FOR K = L TO N
          V(K, J) = V(K, J) + s * V(K, i)
        NEXT K
      NEXT J
    END IF
    FOR J = L TO N
      V(i, J) = 0!
      V(J, i) = 0!
    NEXT J
  END IF
  V(i, i) = 1!
  G = RV1(i)
  L = i
NEXT i

' accumulation of left-hand transformations.
FOR i = N TO 1 STEP -1
  L = i + 1
  G = W(i)
  IF (i < N) THEN
    FOR J = L TO N
      A(i, J) = 0!
    NEXT J
  END IF
  IF (G <> 0!) THEN
    G = 1! / G
    IF (i <> N) THEN
FOR \( J = L \) TO \( N \)
\[
    s = 0!
\]
FOR \( K = L \) TO \( M \)
\[
    s = s + A(K, i) * A(K, J)
\]
NEXT \( K \)
\[
    f = (s / A(i, i)) * G
\]
FOR \( K = i \) TO \( M \)
\[
    A(K, J) = A(K, J) + f * A(K, i)
\]
NEXT \( K \)
NEXT \( J \)
END IF
FOR \( J = i \) TO \( M \)
\[
    A(J, i) = A(J, i) * G
\]
NEXT \( J \)
ELSE
FOR \( J = i \) TO \( M \)
\[
    A(J, i) = 0!
\]
NEXT \( J \)
END IF
\[
    A(i, i) = A(i, i) + 1!
\]
NEXT \( i \)

' diagonalization of the bidiagonal form
' loop over singular values
FOR \( K = N \) TO 1 STEP -1
' loop over allowed iterations
FOR \( I T S = 1 \) TO 30
    FOR \( L = K \) TO 1 STEP -1
' test for splitting
    NM = \( L - 1 \)
' note that \( RV1(1) \) is always zero
    IF ((ABS(RV1(L)) + ANORM) = ANORM) GOTO 2
    IF ((ABS(W(NM)) + ANORM) = ANORM) GOTO 1
    NEXT \( L \)
' cancellation of \( RV1(L) \), if \( L > 1 \)
1
    \( C = 0! \)
    \( s = 1! \)
    FOR \( i = L \) TO \( K \)
        \( f = s * RV1(i) \)
        IF ((ABS(\( f \)) + ANORM) <> ANORM) THEN
            \( G = W(i) \)
            \( H = SQR(f * f + G * G) \)
            \( W(i) = H \)
            \( H = 1! / H \)
            \( C = (G * H) \)
            \( s = -(f * H) \)
            FOR \( J = 1 \) TO \( M \)
                \( y = A(J, NM) \)
                \( A(J, NM) = A(J, NM) + f * A(J, i) \)
            NEXT \( J \)
        END IF
    NEXT \( i \)
    NEXT \( I T S \)
    NEXT \( K \)
NEXT \( I T S \)
END
\[ z = A(J, i) \]
\[ A(J, NM) = (y \times C) + (z \times s) \]
\[ A(J, i) = -(y \times s) + (z \times C) \]

\[ \text{NEXT } J \]
\[ \text{END IF} \]
\[ \text{NEXT } i \]

2 \[ z = W(K) \]

' convergence

IF (L = K) THEN

' singular value is made nonnegative.

IF (z < 0!) THEN

\[ W(K) = -z \]
\[ \text{FOR } J = 1 \text{ TO } N \]
\[ V(J, K) = -V(J, K) \]
\[ \text{NEXT } J \]
\[ \text{END IF} \]
\[ \text{GO TO } 3 \]
\[ \text{END IF} \]

IF (ITR = 30) THEN PRINT 'No converge in 30 iterations'

' shift from bottom to 2-by-2 minor

\[ X = W(L) \]
\[ NM = K - 1 \]
\[ y = W(NM) \]
\[ G = RV1(NM) \]
\[ H = RV1(K) \]
\[ f = ((y - z) \times (y + z) + (G - H) \times (G + H)) / (2! \times H \times y) \]
\[ G = \text{SQRT}(f \times f + 1!) \]
\[ zzz = (X - z) \times (X + z) \]
\[ f = (zzz + H \times ((y / (f + \text{ABS}(G) \times \text{SGN}(f))) - H)) / X \]

' next QR transformation

\[ C = 1! \]
\[ s = 1! \]
\[ \text{FOR } J = L \text{ TO } NM \]
\[ i = J + 1 \]
\[ G = RV1(i) \]
\[ y = W(i) \]
\[ H = s \times G \]
\[ G = C \times G \]
\[ z = \text{SQRT}(f \times f + H \times H) \]
\[ RV1(J) = z \]
\[ C = f / z \]
\[ s = H / z \]
\[ f = (X \times C) + (G \times s) \]
\[ G = -(X \times s) + (G \times C) \]
\[ H = y \times s \]
\[ y = y \times C \]
\[ \text{FOR NM = 1 \text{ TO } N} \]
\[ \begin{align*}
X &= V(NM, J) \\
z &= V(NM, i) \\
V(NM, J) &= (X \times C) + (z \times s) \\
V(NM, i) &= -(X \times s) + (z \times C) \\
\end{align*} \]

\( z = \text{SQRT}(f \ast f + H \ast H) \)

' rotation can be arbitrary if \( Z=0 \)
\[ \begin{align*}
W(J) &= z \\
\text{IF} (z <> 0!) \text{ THEN} \\
z &= 1!/z \\
C &= f \ast z \\
s &= H \ast z \\
\end{align*} \]

\( f = (C \ast G) + (s \ast y) \\
X = -(s \ast G) + (C \ast y) \)

\( \text{FOR} \ NM = 1 \ \text{TO} \ M \\
y &= A(NM, J) \\
z &= A(NM, i) \\
A(NM, J) &= (y \ast C) + (z \ast s) \\
A(NM, i) &= -(y \ast s) + (z \ast C) \\
\end{align*} \]

\( \text{NEXT} \ NM \)
\( \text{NEXT} \ J \)
\( RV1(L) = 0! \)
\( RV1(K) = f \)
\( W(K) = X \)
\( \text{NEXT} \ ITS \)
\( 3 \)
\( \text{NOP} \ 	ext{statement} \)
\( \text{NEXT} \ K \)

\( \text{END} \ \text{SUB} \)

\text{SUB} SVDFIT (X(), y(), sig(), ND\text{DATA}, A(), MA, U(), V(), W(), MP, NP, cq)

' From "Numerical Recipes", converted to BASIC by "FORT2BAS.BAS"

' given a set of ND\text{DATA} points \( X(I), Y(I) \) with individual standard
' deviations SIG(I), use Chi-square minimization to determine the MA
' coefficients A of the fitting function
' \[ y = \text{SUM-I} (A-sub-I \ast \text{AFUNC-sub-I}(x)). \]
' Here we solve the fitting equations using singular value
' decomposition of the ND\text{DATA} by MA matrix. Arrays U, V, W, provide
' workspace on input, on output they define the singular value
' decomposition, and can be used to obtain the covariance matrix.
MP, NP are the physical dimensions of the matrices U, V, W as indicated below. It is necessary that MP > NDATA, and NP >= MA.
The program returns values for the MA fit parameters A, and Chi-square CHISQ. The user supplies a subroutine
FUNCS(X(i), AFUNC(), MA) that returns the MA basis functions evaluated at x = X in the array AFUNC.

FUNCS is the function subprogram

```
DIM AFUNC(max), b(nmax)
' accumulate coefficients of the fitting matrix.
FOR i = 1 TO NDATA
    CALL funcs(X(i), AFUNC(), MA)
    tmp = 1 / sig(i)
    FOR J = 1 TO MA
        U(i, J) = AFUNC(J) * tmp
    NEXT J
    b(i) = y(i) * tmp
NEXT i
' singular value decomposition
CALL SVDCMP(U<), NDATA, MA, MP, NP, W(), V())
WMAX = 0
' edit the singular values, given TOL . from here ....

FOR J = 1 TO MA
    IF (W(J) > WMAX) THEN WMAX = W(J)
NEXT J

THRESH = tol * WMAX
FOR J = 1 TO MA
    IF (W(J) < THRESH) THEN W(J) = 0!
NEXT J
' to here .......
CALL SVGKSB(U<), W(), V(), NDATA, MA, MP, NP, b(), A())

chisq = 0!
' evaluate chi-square
FOR i = 1 TO NDATA
    CALL funcs(X(i), AFUNC(), MA)
    SUM = 0!
    FOR J = 1 TO MA
        SUM = SUM + A(J) * AFUNC(J)
    NEXT J
    chisq = chisq + ((y(i) - SUM) / sig(i))^2
```
SUB SVDVAR (V(), MA, NP, W(), CVM(), ncvm)
'-------------------------------------------------------------
' From "Numerical Recipes", converted to BASIC by "FORT2BAS.BAS"
'
' to evaluate the covariance matrix CVM of the fit for MA parameters
' obtained by SVDFIT, call this routine with matrices V, W as
' returned from SVDFIT. NP, NCVM give the physical dimensions of
' V, W, CVM as indicated below.
'-------------------------------------------------------------

DIM WTI(mmax)

FOR i = 1 TO CINT(MA)
    WTI(i) = 0!
    IF (W(i) <> 0!) THEN WTI(i) = 1! / (W(i) * W(i))
NEXT i

' sum contributions to covariance matrix
FOR i = 1 TO CINT(MA)
    FOR J = 1 TO i
        SUM = 0!
        FOR K = 1 TO CINT(MA)
            SUM = SUM + V(i, K) * V(J, K) * WTI(K)
        NEXT K
        CVM(i, J) = SUM
        CVM(J, i) = SUM
    NEXT J
NEXT i

END SUB
1. H. K. Onnes, Comm. from the Univ. of Leiden, **122**, 81(1911).


5. J. G. Bednorz, Ibid.


11. C. W. Chu, Ibid.


19. Willis, et. al., Ibid.


21. Willis, Ibid.


26. C. L. Chien, Ibid.


39. Samples were prepared by Hyunsu Shin, and Ying-Chao Wang.


45. Callaway, J., Ibid.
48. Q. Li et. al., Ibid.
50. Peter A. Stark, Ibid., pp. 95-99.
51. Q. Li, Ibid.
52. Brown, et. al., Ibid.
55. Brown, Ibid.
57. Mattis, Ibid.
61. Samples prepared by Hyunsu Shin and Ying-Chao Wang.
62. J. O. Willis, et al., Ibid.

66. Morrish, Ibid., pg. 263.


68. J. O. Willis, et al., Ibid.


71. C. L. Smith, Ibid.


Carter H. Watson    S.S. Number 439-15-5484

LSU : Dept. of Physics LSU, Baton Rouge, La 70803    Ph.: (504)-388-5859
Perm : 1502 Bellaire Blvd. Bossier City, La 71112    Ph.: (318)-742-1282

Education:

LSU-BR  M.S.E.E. Fall 1986  G.P.A. 3.9
University of Mississippi M.S. Physics Summer 1980
LSU-Sh  B.S. Physics Spring 1979
LSU-Sh  B.S. Math Spring 1978 - Magna Cum Laude

Experience:

Lab Assistant, Dept. of Physics, LSU-Sh 1977-79
Lab Assistant, Dept. of Nuclear Medicine, LSU-Sh School of Medicine, summer 1978. (mainly handling and disposal of nuclear drugs and nuclear waste; also radiation safety)
Research/Teaching Asst., Dept. of Physics, Univ. of Mississippi, 1979-81 (member of tornadic storm intercept crew interested in obtaining electric, acoustic, and photographic information on severe storms in association with National Severe Storms Laboratory, Norman, Ok. Was crew-chief in 1981)
Teaching Asst., Dept. of Physics, LSU-BR, Fall 1981 - Summer 1988
Research Asst. (no pay), Dept. of Physics, LSU-BR, Summer 1982 - present. (mainly Solid-State experimental physics of organic and High Tc superconductors). This includes evaluation and purchasing decisions for the lab.

Capabilities:

Fluent in a number of computer languages.
Use several different CAD systems in course of work.
Have automated several of our solid-state experiments (small-sample low-temp specific heat, low-temp R.F. surface impedance, several one-shot experiments).
Competent numerical analyst.
Have designed, constructed, debugged, and modified many instruments in order to achieve necessary measurements (analog and digital) while under constraints (time, money, etc.)
Have recommended and selected equipment (MCA's, micro-computer data collection systems, LIA's, software, etc.) for the labs.
Have supervised field data collection teams, and undergraduate research projects.
Handling of vacuum systems (construction, Mass-spec leak chasing, etc.)
Handling of cryogenic fluids.
Competent in machine shop.

Achievements:
Six publications, two pending
Member of Phi-Kappa-Phi, Alpha-Sigma-Omicron, Sigma-Pi-Sigma (all are elective national honor societies)
Presented papers at APS/SPS meetings in Houston and Biloxi.
Attended the NATO-ASI on Organic Superconductors in Magog, Quebec Aug-Sept 1986

Interests:
Recreational computing; all forms of electronics; car and home repair/improvement; astronomy; racquet sports and jogging

Personal:
Born 9/12/57 Topeka, Kansas.
Married 6/27/79 to Dr. Ann Branch, Shreveport, La.

References:
Dr. Paul Kirk, Dept of Physics, LSU-Baton Rouge, La, 70803
Dr. Roy Goodrich, Dept of Physics, LSU-Baton Rouge, La, 70803,
   Phone: (504)-388-5847 for both of above
Dr. Ben George, Saraland Veterinary Clinic, Hwy 43, Saraland, Al, 36571
   Phone: (205)-679-0370
Publications:


"Specific Heat and Magnetic Order in GdBa₂Sr₂Cu₃O₇," C. H. Watson, D. A. Brown, Jing-chun Xu, R. G. Goodrich; Physical Review (submitted)

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Carter Harrison Watson

Major Field: Physics

Title of Dissertation: Small Sample Calorimetry and Specific-Heat of \( \text{Gd}_1\text{Ba}_{2-x}\text{Sr}_x\text{Cu}_3\text{O}_7 \)

Approved:

Roy C. Goodrich
Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

John P. Weibel

Lai-Tak Chan

Danny D. Reade

Robert F. Lax

Date of Examination: 24 July 1989