1988

Star Formation, Using Three-Dimensional Explicit Eulerian Hydrodynamics.

Harold Alden Williams
Louisiana State University and Agricultural & Mechanical College

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

Recommended Citation
https://digitalcommons.lsu.edu/gradschool_disstheses/4549

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 original text directly from the copy submitted. Thus, some dissertation copies are in typewriter face, while others may be from a computer printer.

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 copyrighted 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 oversize page is available as one exposure on a standard 35 mm slide or as a 17" × 23" black and white photographic print for an additional charge.

Photographs included in the original manuscript have been reproduced xerographically in this copy. 35 mm slides or 6" × 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.
Star formation, using 3-D explicit Eulerian hydrodynamics

Williams, Harold Alden, Ph.D.
The Louisiana State University and Agricultural and Mechanical Col., 1988
PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark ✓.

1. Glossy photographs or pages _____
2. Colored illustrations, paper or print _____
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy _____
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages ✓
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) __________ lacking when material received, and not available from school or author.
12. Page(s) __________ seem to be missing in numbering only as text follows.
13. Two pages numbered ______. Text follows.
14. Curling and wrinkled pages _____
15. Dissertation contains pages with print at a slant, filmed as received ______
16. Other ________________________________

____________________________________
____________________________________

UMI
Star Formation, Using 3-D Explicit Eulerian Hydrodynamics

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
Harold Alden Williams
B.S., Florida State University, 1973
M.S., State University of New York at Stony Brook, 1979
May, 1988
Acknowledgments

I acknowledge and thank the following persons. Joel Tohline taught me the art of astrophysical hydrodynamics. Dick Durisen and Izumi Hachisu have provided axisymmetric hydrostatic equilibria. Both Dick and Izumi have increased my understanding of hydrodynamics. Dimitris Christodoulou has been a friend and sounding board for my ideas and has shared Joel’s time with me. Norman Lebovitz has provided the vision of another possible way to achieve binary fission, unfortunately hydrostatic equilibria in this region of the model space have not yet been produced. Alan Boss is providing an exciting environment for work after I leave here—radiation transport, here I come.

Barbara and Stephanie have loved me. My parents raised me. My teachers taught me. Monetarily, the National Science Foundation has supported some of my work (through grants AST-8501842 and AST-8701503); the state of Louisiana, through Louisiana State University, has supported the rest. The people at the System Network Computer Center, and physics and astronomy department staff members Monika Lee and Hortensia Valdes, have helped me compute.

I also acknowledge a fortunate accident: Dick Haymaker’s assigning me an astronomy lab to teach after I asked the question, “Are you offering any graduate courses in astronomy this semester?”
Preface

This dissertation is unusual in that it does not cover a single well-defined problem but instead describes an art, astrophysical hydrodynamic computer code writing. Appendix A, a reprint of “Linear and Nonlinear Dynamic Instability of Rotating Polytropes” (Williams and Tohline 1987 a), is a well-defined problem and its solution. So is Appendix B, a preprint of “Circumstellar Ring Formation in Rapidly Rotating Protostars” (Williams and Tohline 1987 d). The art, the writing and running of a computer code in astrophysical hydrodynamics, is more general and ultimately more useful than either of the narrowly focused problems. The art allows many questions to be answered. The model space is astronomically large, in every sense of the word. After all, it attempts to fulfill the Newtonian boast, given the velocity and mass density at one instant in time to predict what they will be at a much later time.

This is not a dissertation on computer algorithms, but Chapter 3, “The Computer Code Itself,” and Appendix C, “Second-Order 3-D Explicit Eulerian Hydrodynamic Computer Code List in FORTRAN,” describe in some detail what was actually done and give a concrete example. If this were an experimental thesis these sections would be the description of the apparatus. Given a FORTRAN compiler and a fairly large computer, Appendix C is the apparatus.

Chapter 2, “Hydrodynamic Equations,” describes the algebraic structure of the equations and how one actually makes the approximation in going from a continuous to a discrete system. Some of the art of numerical approximation and numerical stability is explained. I am most proud of Chapter 2 in that it shows my mastery over the subject.
Chapter 4, “Analyzing Data,” describes the formidable task of making sense out of all of the numbers that result from this type of numerical simulation. The results in 3-D hydrodynamics are generally very difficult to interpret — after all, there are all these 3-D arrays evolved in time, an embarrassment of riches. The data must be imaged to make interpretation possible. At some stage in our investigation of astrophysical hydrodynamics, I hope that images taken at a telescope — at optical, infrared, or whatever wavelength — can be compared to the hydrodynamic simulation. Some of the imaging software used was developed for radio astronomy, AIPS; and other parts of the software were developed for computer assisted tomography (CAT scans), to diagnose disease in the human body. The most useful of all were the widely available graphics routines from the National Center for Atmospheric Research, NCAR.

Chapter 5, “Models Run in the Second-Order Code, a Description and Preliminary Conclusions,” details some of the models that have been run using the second-order code, as opposed to the results reported in Appendices A and B, which were run using the first-order code. When I am sure that I have understood the results and that they are most probably correct, they will be prepared for publication in a manner similar to Appendices A and B. One of the models in Chapter 5 was a verification run where the answer was known, but the rest are new models where the answers are new.

Chapter 6, “Future Directions,” describes some of my research plans. This being an art, and not just a dissertation, there is much to be done.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
# Table of Contents

Acknowledgments ....................................................................................................... ii  
Preface ........................................................................................................................ iii  
Table of Contents ....................................................................................................... v  
List of Tables ........................................................................................................... viii  
List of Figures ........................................................................................................... ix  
Abstract ..................................................................................................................... xiii  

1 Introduction ............................................................................................................. 1  

2 Hydrodynamic Equations ....................................................................................... 8  
2.1 Basic Equations and Choices ........................................................................... 8  
2.2 Conservative Form ......................................................................................... 14  
2.3 Sourcing and Fluxing ...................................................................................... 18  
2.4 Sourcing and Fluxing Separately .................................................................. 24  
2.5 First-Order Fluxing, Donor Cell .................................................................. 26  
2.6 Second-Order Fluxing, Van Leer Monotonic Interpolation ......................... 27  
2.7 Consistent Advection .................................................................................... 30  
2.8 Errors Caused by Interpolation .................................................................... 33  
2.9 Viscosity ............................................................................................................ 35  
2.10 Courant Limit .................................................................................................. 39  

3 The Computer Code Itself ..................................................................................... 41  
3.1 A Flow Chart in Words .................................................................................. 41  
3.2 Computers Today ............................................................................................ 47  
3.3 Conditional Vector Merge and Bit Mask Registers .................................. 50  

4 Analyzing Data ....................................................................................................... 53  
4.1 Graph Everything ............................................................................................ 53  
4.2 Fourier Analysis and Spectral Methods ....................................................... 55  
4.3 Image Everything ............................................................................................. 58  

5 Models Run in the Second-Order Code: Description and Preliminary  
Conclusions ............................................................................................................. 62  
5.1 General Description ....................................................................................... 62
5.2 Model with $n = 1.5, \beta_i = 8.9 \times 10^{-5}, q = 3/2$ .................. 67
5.3 Model with $n = 1.8, \beta_i = 0.313, q = 3/2$ .......................... 72

6 Summary .................................................................................. 78
6.1 Gravitational-Rotational Dynamic Instability ...................... 78
6.2 Second-Order Hydrodynamics ............................................... 80

7 Future Directions ................................................................. 82
7.1 Different Rotation Laws ......................................................... 82
7.2 White Dwarf Equations of State ........................................... 83
7.3 Gray Radiation Transport ..................................................... 83
7.4 Cartesian Hydrodynamics, using HW3CRT to solve $\nabla^2 \Phi = 4\pi G \rho$ ................................ 83
7.5 Adding more Source Terms, Putting in More Physics ........... 84

References .................................................................................. 86

Appendices
A Linear and Nonlinear Dynamic Instability of Rotating
Polytropes ................................................................. 96
  Abstract ..................................................................... 97
  I. Introduction .......................................................... 97
  II. Model Description .................................................. 97
  III. Analysis ............................................................... 98
  IV. Evolutions ............................................................ 99
  V. Conclusions ............................................................ 100
  Appendix Polytropic Units to Other Units ..................... 103

B Circumstellar Ring Formation in Rapidly Rotating Protostars ... 105
  Abstract ...................................................................... 105
  I. Introduction .......................................................... 106
  II. Definition of the Problem ........................................ 110
    a) Initial Models .................................................... 110
    b) Physics and Solution Method ............................... 115
  III. Evolutions .......................................................... 120
  IV. Discussion and Conclusions .................................. 133

vi

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
a) General Summary ............................................. 133
b) Application to Protostars ................................. 137
c) Other Astrophysical Systems ............................ 141
d) Future Work ..................................................... 143

C Second-Order 3-D Explicit Eulerian Hydrodynamic Computer
Code List in FORTRAN ....................................... 147
C.1 Introduction ................................................... 147
C.2 MAININ ....................................................... 150
C.3 MAINCN ....................................................... 172
C.4 HYD3D2O .................................................... 189
C.5 POTS .......................................................... 226
Vita ................................................................. 250
List of Tables

Appendix A

Table 1 Initial Polytrope Structure ................................................. 98
Table 2 Hydrodynamic and TVE Results ........................................... 99

Appendix B

Table 1 Initial Polytrope Structure ................................................... 146
Table B.2 Hydrodynamic Results ..................................................... 146
Table 5.3 Results of Fission Simulations .......................................... 146
List of Figures

Fig. 1.1 log T versus log ρ ................................................................. 4

Fig. 2.1 Control volume r ............................................................... 21

Fig. 4.1 STAR4 black-and-white image .......................... 61

Fig. 5.1 Graph of ln D versus polytropic time for the n = 1.8, βi = 0.310 model at r/Req = 0.640, calculated with Tohline’s first-order code ..................... 64

Fig. 5.2 Graph of ln D versus polytropic time for the n = 1.8, βi = 0.310 model at r/Req = 0.640, calculated with my second-order code ..................... 65

Fig. 5.3 Graph of ln D versus polytropic time for the n = 1.5, βi = 8.9 × 10^{-5} model at r/Req = 0.640, calculated with my second-order code when fluxing and sourcing were done together .................................................. 68

Fig. 5.4 Graph of ln D versus polytropic time for the n = 1.5, βi = 8.9 × 10^{-5} model at r/Req = 0.640, calculated with my second-order code when fluxing and sourcing were done together for S and T, but separately for A. ................. 69

Fig. 5.5 Contour plot of ρ(r, z = 0, θ) for the n = 1.5, βi = 8.9 × 10^{-5} model at 20.74 polytropic time, calculated with my second-order code when fluxing and sourcing were done together for S and T, but separately for A. ................. 70

Fig. 5.6 Contour plot of ρ(r, z, θ = 0) for the n = 1.5, βi = 8.9 × 10^{-5} model at 20.74 polytropic time, calculated with my second-order code when fluxing and sourcing were done together for S and T, but separately for A. ................. 71

Fig. 5.7 Graph of ln D versus polytropic time for the n = 1.5, βi = 8.9 × 10^{-5} model at r/Req = 0.640, calculated with my second-order code when sourcing
and fluxing were done separately for all dynamical variables. ........... 73

Fig. 5.8 Contour plot of $\rho(r, z = 0, \theta)$ for the $n = 1.5, \beta_i = 8.9 \times 10^{-5}$ model at 20.74 polytropic time, calculated with my second-order code when sourcing and fluxing were done separately for all dynamical variables. ................. 74

Fig. 5.9 Contour plot of $\rho(r, z, \theta = 0)$ for the $n = 1.5, \beta_i = 8.9 \times 10^{-5}$ model at 20.74 polytropic time, calculated with my second-order code when sourcing and fluxing were done separately for all dynamical variables ................. 75

Fig. 5.10 Graph of $\ln D$ versus time in central rotation periods for the $n = 1.8, \beta_i = 0.313$ model at $r/R_{eq} = 0.640$, calculated with my second-order code when sourcing and fluxing were done separately for all dynamical variables .... 77

Appendix A

Fig. 1 Graph of $\ln D$ versus polytropic time for the $n = 1.8, \beta_i = 0.310$ model at $r/R_{eq} = 0.64$ ................................................................. 99

Fig. 2 Graph of the phase angle $\phi_m$ for three different Fourier modes versus polytropic time for the $n = 1.8, \beta_i = 0.310$ model at $r/R_{eq} = 0.64$ ............ 100

Fig. 3 Graph of the ratio of pattern frequencies for several Fourier modes versus polytropic time for the $n = 1.8, \beta_i = 0.310$ model at $r/R_{eq} = 0.64$ ............ 100

Fig. 4 A 3-D surface of $\ln D_2$ versus $r/R_{eq}$ and polytropic time for the $n = 1.8, \beta_i = 0.310$ model ......................................................... 101

Fig. 5 A 3-D surface of $\phi_2/(2\pi)$ versus $r/R_{eq}$ and polytropic time for the $n = 1.8, \beta_i = 0.310$ model ......................................................... 101

Fig. 6 An isodensity surface for a typical initial model of $\beta_i = 0.310$; actually for

x
the $n = 1.3$ model .................................................................................................................. 101
Fig. 7 Isodensity surfaces for the $n = 0.8$ and $\beta_i = 0.310$ model with polytropic
time labeling the surfaces ........................................................................................................ 101
Fig. 8 Isodensity surfaces for the $n = 1.0$ and $\beta_i = 0.310$ model with polytropic
time labeling the surfaces ........................................................................................................ 102
Fig. 9 Isodensity surfaces for the $n = 1.3$ and $\beta_i = 0.310$ model with polytropic
time labeling the surfaces ........................................................................................................ 102
Fig. 10 Isodensity surfaces for the $n = 1.5$ and $\beta_i = 0.310$ model with polytropic
time labeling the surfaces ........................................................................................................ 102
Fig. 11 Isodensity surfaces for the $n = 1.8$ and $\beta_i = 0.310$ model with polytropic
time labeling the surfaces ........................................................................................................ 102

Appendix B

Fig. 1 Some normalized frequencies versus $r/R_{eq}$ for the initial $n = 1.8$, $\beta_i = 0.310$
model ....................................................................................................................................... 116
Fig. 2 Some normalized frequencies versus $r/R_{eq}$ for the initial $n = 0.8$, $\beta_i = 0.310$
model ....................................................................................................................................... 117
Fig. 3 Isodensity surfaces for the $n = 1.8$ and $\beta_i = 0.310$ model with central
rotation period times labeling the surfaces ............................................................................... 122
Fig. 4 Isodensity surfaces for the $n = 0.8$ and $\beta_i = 0.310$ model with central
rotation period times labeling the surfaces ............................................................................... 123
Fig. 5 Different $\rho/\rho_c$ isodensity surfaces for the $n = 1.8$ and $\beta_i = 0.310$ model at
12.21 central rotation periods ................................................................................................. 124
Fig. 6 Different $\rho/\rho_c$ isodensity surfaces for the $n = 1.8$ and $\beta_i = 0.310$ model at 19.4 central rotation periods ................................................................. 125

Fig. 7 Four graphs showing the behavior of certain dynamical variables versus time in central rotation periods. (a) Pattern speed only for the $n = 1.8$ and $\beta_i = 0.310$ model at $r/R_{eq} = 0.64$. For both the $n = 1.8$ and $n = 0.8$ models: (b) $\beta$, (c) $J(t)/J(t = 0)$; (d) Mass ....................................................................................... 128

Fig. 8 Two graphs illustrating the time-variation of two variables in the central region (at $r/R_{eq} = 0.24$) of the $n = 1.8$ model. (a) Pattern speed; (b) $\ln D_2$ 129

Fig. 9 A 3-D surface of $\ln(C_0/2)$ versus $r/R_{eq}$ and time in central rotation periods for the $n = 1.8$, $\beta_i = 0.310$ model ................................................................. 132
Abstract

Evolutions of rapidly rotating, self-gravitating objects initially in axisymmetric equilibrium have been studied using a 3-D Newtonian hydrodynamic computer code with an eye toward understanding angular momentum transport in dynamically evolving protostars. First, a number of evolutions have been modeled using an existing explicit, Eulerian, finite difference code that is accurate to first-order in its spatial differences. The bar-mode dynamic instability at has been explored by considering several models with different degrees of compressibility. This instability occurs in models having $\beta > \beta_d \equiv 0.27$, where $\beta$ is the ratio of the rotational to the gravitational potential energy. A two-armed spiral, with a well-defined pattern speed and growth rate that match the pattern speed and growth rate predicted by linear theory, develops from each of the axisymmetric equilibria. The models with greater compressibility exhibit spirals which are more tightly wound. As the nonaxisymmetric distortion becomes large in an extended evolution, the object does not undergo binary fission as had been thought earlier. Instead, the spiral elongates and then wraps up on itself, forming a central pulsating triaxial object surrounded by a more diffuse “ring-like” disk. Angular momentum and mass are dynamically redistributed by gravitational torques during the evolution, and $\beta$ is reduced below $\beta_d$. Since this gravitational-rotational dynamic instability is a general feature of gaseous systems, this study may have application to theta galaxies and to rapidly rotating neutron stars, as well as to protostars.

The hydrodynamic code has been modified to include second-order-accurate spatial differences and has been rewritten so that it now vectorizes well. The
results obtained from running a model in the second-order code are qualitatively similar to the results obtained from the first-order code. Quantitatively, however, the answers are somewhat different and are, presumably, more reliable. In the second-order code, growth rates no longer need to be corrected for the effects of numerical viscosity to agree with the predictions of linear theory. Higher order nonaxisymmetric Fourier modes are also no longer damped.
1. INTRODUCTION

Stars form out of the interstellar medium, whose neutral atomic hydrogen component has a typical density of $3 \times 10^{-23} \text{g cm}^{-3}$ and a temperature around 80K. When stars reach the hydrogen-burning main sequence, they have a mean density of around $1 \text{g cm}^{-3}$ and a mean temperature of several million degrees Kelvin. The size of an interstellar gas cloud is around $10^{19} \text{cm}$, and the size of the resulting star is typically $10^{11} \text{cm}$. The specific angular momentum of star-forming regions is around $10^{23} \text{cm}^2 \text{s}^{-1}$; for T Tauri stars (which are young stars) it is around $10^{17} \text{cm}^2 \text{s}^{-1}$; and for the 4.6 billion-year-old Sun, it is now $10^{15} \text{cm}^2 \text{s}^{-1}$ (Bodenheimer 1983). Clearly, any theory that attempts to explain the process of star formation must explain how, during its gravitational contraction over many orders of magnitude in density and radius, a gas cloud loses or redistributes a substantial fraction of its angular momentum. Analytic methods provide only a limited amount of information on this evolution, and significant progress toward a solution to the problem of star formation must, by necessity, incorporate numerical tools.

The end point of this evolution can be observed. We see stars in the optical part of the electromagnetic spectrum when they reach the main sequence because they typically have a surface temperature of thousands of degrees Kelvin, which puts the peak emission of their continuous, black-body radiation emission in the optical. The early evolution of a contracting gas cloud, from densities $\sim 10^{-22} \text{g cm}^{-3}$ to $\sim 10^{-16} \text{g cm}^{-3}$, can be directly observed in millimeter wavelength line radiation from molecules like CO, H$_2$O, and NH$_3$. These molecules exist in trace amounts in H$_2$ clouds, and their millimeter wavelength line radiation is excited.
only at low densities. Infrared radiation from dust grains can be seen, but infrared radiation becomes optically thick at about $10^{-14}$ g cm$^{-3}$. The forming molecular cloud cores cannot be observed at densities higher than this, since they then have infrared photospheres. At present, direct observations probing the evolution between densities $\sim 10^{-14}$ and the final stellar state (i.e., over the last 14 orders of magnitude in density!) do not exist, so this part of the evolution can only be simulated.

Figure 1.1 is a generic log $T$ versus log $\rho$ graph covering the parameter range relevant to star formation, where $T$ is the interior temperature in degrees Kelvin and $\rho$ the interior mass density in g cm$^{-3}$. Diagrams similar to this are in Bodenheimer (1978, 1983), Winkler and Newman (1980), Larson (1985), and Boss (1985). The solid curve drawn in this diagram, derived using 1-D hydrodynamic (spherical symmetry) simulations and including all of the relevant physics, depicts the evolutionary path that a one-solar-mass gas cloud should follow during its collapse to the stellar state. From left to right along the curve: first is a cooling phase, (during which an initially atomic cloud becomes predominantly molecular); then a flat isothermal phase; then a heating phase; then the slope drops somewhat when molecular hydrogen dissociates ($H_2 \rightarrow 2H$); and after all of the hydrogen is in atomic form, the cloud heats up more rapidly again. The cooling and isothermal phases are the first dynamic collapse phases: this is when fragmentation of a large-mass gas cloud into smaller clumps of normal stellar mass is generally thought to occur. This region is observable by millimeter-wave radio telescopes. The first heating phase provides an opportunity for a first quasi-equilibrium phase of contraction to exist. The dissociation phase identifies
a second dynamic collapse phase. Notice that, during the dissociation phase, the densities are such that the free-fall time is very short, only a few years. The free-fall time is $t_{ff} = [3\pi/(32G\rho)]^{1/2}$ and this is the time it takes for a pressure-free, uniformly dense ball of density $\rho$ to homologously free-fall. After dissociation is completed rapid heating results in a second quasi-equilibrium contraction phase.

During the first dynamic collapse phase, magnetic fields are almost certainly important in most star-forming regions and no doubt remove some of the excess specific angular momentum from a cloud (Mouschovias 1977, 1978, 1981). As a cloud enters the first quasi-equilibrium phase of contraction, however, the density is large enough so that the fractional ionization of the gas is extremely small, so the magnetic field has nothing to work with and diffuses out of the cloud. From that point on, magnetic fields are probably unimportant to the gross dynamics of a typical star-forming cloud.

During both of the quasi-equilibrium phases, the virial theorem demands that $\alpha + \beta = \frac{1}{2}$, where $\alpha$ is the ratio of thermal to gravitational potential energy $E_{\text{thermal}}/|E_{\text{gra}}|$ and $\beta$ is the ratio of rotational to gravitational potential energy $E_{\text{rot}}/|E_{\text{gra}}|$. Since there can still be an abundance of angular momentum in a cloud at this stage, with the specific angular momentum being much higher at the beginning of the first quasi-equilibrium phase than in the final stellar state, it is likely that many clouds will possess a nonnegligible rotational energy (have $\beta$ larger than $\alpha$)—some certainly do.

To simulate a rapidly rotating, quasi-equilibrium phase of a protostar's evolution requires a 3-D hydrodynamic treatment since none of the space variables can be ignored. This is one reason for developing a computer code that will
Figure 1.1.—Evolution of the interior temperature and density for a spherical protostar. Also shown is the gravitational free-fall collapse time $t_{ff} = [3\pi/(32G\rho)]^{1/2}$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
perform 3-D explicit Eulerian hydrodynamics.

It is well known that as the parameter $\beta$ is increased, the first nonaxisymmetric dynamic instabilities occur at $\beta = \beta_d \equiv 0.27$ in self-gravitating, uniformly rotating, incompressible objects (Lyttelton 1953, Chandrasekhar 1969, and Tassoul 1978). Of course, real gas clouds are compressible. Furthermore, the analyses that have predicted $\beta_d$, the so called bar-mode instability, are valid only in the limit of small distortions from axisymmetric equilibria. Nonlinear hydrodynamic techniques for studying compressible gas flows are required to learn more about this instability in realistic systems. As a further enticement to investigate this instability, it has been proposed that the instability might provide a way to form close binary stars (see Durisen and Tohline 1985 for a review). The dynamic instability would be expected to grow to nonlinear amplitudes and lead to binary fission. Binary fission is potentially very important since it is now well established that at least 50% of all stars are in close binary pairs, despite the recent controversy connected with exactly which fraction of the binaries are at what period (Abt 1987 and references therein).

With this motivation in mind, five axisymmetric equilibria with different degrees of compressibility were evolved to slightly nonlinear amplitudes using a first-order, 3-D explicit Eulerian hydrocode written by Tohline (1978). The complete results of this work are provided in Appendix A, a paper that I coauthored with Tohline and published in The Astrophysical Journal in April, 1987. The results, besides confirming the predictions of linear theory for the growth rate and pattern speed of the nonaxisymmetric bar-mode, are that this instability leads to a spiral mode. The spiral is more tightly wound with increased compressibility.
The properties of the spiral-mode eigenfunction were more carefully determined in this study than had been done in any previous study.

Two of these models were further evolved, and a circumstellar ring around a central, pulsating, triaxial object was observed to be the final outcome of the dynamical bar-mode, or "fission," instability. This work is explained in Appendix B, a paper which has been submitted for publication in *The Astrophysical Journal*. In the nonlinear evolution the pattern speed increased as the spiral arms elongated and then wrapped up on themselves. A circumstellar ring forms around a more centrally condensed triaxial figure that is rotating about its shortest axis. Angular momentum is transferred outward and $\beta$ is reduced below $\beta_d$. These simulations used the lowest value of $\beta > \beta_d$ so far. A slowly contracting protostar will never evolve to a configuration in which $\beta$ is substantially larger than $\beta_d$ if it undergoes a dynamical instability at or near $\beta_d$ that results in redistribution of angular momentum and a corresponding reduction in the value of $\beta$.

Even as the first simulations were being done, it became apparent that Tohline's first-order code would have to be improved in order to handle lower values of $\beta$ because the numerical diffusion in a first-order code would keep the nonaxisymmetric instability from developing at all in models that were only marginally unstable. The nonlinear behavior of this instability may also depend upon the development of higher order modes, and these are even more strongly damped than the "bar" or two-armed spiral mode in a first-order code. In order to overcome these numerical diffusion problems, a second-order code was written and tested by me.

Chapter 2 is a description of the hydrodynamic equations and a detailed de-
scription of how they are solved explicitly in time and to second-order spatially. Chapter 3 gives a guide to the subroutine names and how they implement the equations discussed in Chapter 2, while Appendix C provides a list of the actual FORTRAN code; Chapter 3 and Appendix C are designed to help anyone who would like to understand the code so that he can modify it. The utility of conditional vector merge functions on vector processors is also discussed in Chapter 3. This has been included because the time demands of a numerical code of this scale virtually demand the use of vectorized algorithms. Chapter 4 discusses how models are analyzed. Of course, good examples of how one analyzes hydrodynamic simulations are the self contained papers in Appendices A and B. Chapter 5 is a sampling of some of the models I have run in the second-order code. The second-order code performs as expected; in the linear amplitude regime, the growth rates are very close to the ones predicted by linear theory. No compensation for numerical diffusion is necessary, as it is in the first-order code. The isodensity surfaces in the simulations run thus far are indistinguishable from one another in the first- or second-order code. Delicate interplay between higher Fourier modes is revealed in the second-order code. In the linear regime, when the axisymmetric distortion is small, the higher order Fourier modes are not damped as they are in the first-order code. Chapter 6 is a description of the research I plan to do in the next few years. Since I have accepted a postdoctoral position at the Department of Terrestrial Magnetism, Carnegie Institution of Washington, working with Alan Boss, there is every reason to believe that I can complete some or all of these proposed projects in the near future.
2. HYDRODYNAMIC EQUATIONS

2.1 BASIC EQUATIONS AND CHOICES

The principal equations governing Newtonian hydrodynamics are the equation of continuity, the equation of motion, and the thermodynamic energy equation. These equations can all be solved by the same method. These equations come from simple principles. The equation of continuity is just a statement of conservation of mass:

\[ \frac{d\rho}{dt} + \rho \nabla \cdot \mathbf{v} = 0, \quad (2.1) \]

where \( \rho \) is mass density, \( t \) is time, and \( \mathbf{v} \) is the fluid velocity. The equation of motion is just Newton's second law—a statement of conservation of momentum—written here per unit volume:

\[ \rho \frac{d\mathbf{v}}{dt} = \mathbf{f}, \quad (2.2) \]

where \( \mathbf{f} \) is the force per unit volume. The energy equation is just a statement of conservation of energy, the first law of thermodynamics:

\[ \frac{d\varepsilon}{dt} + \varepsilon \nabla \cdot \mathbf{v} = -P \nabla \cdot \mathbf{v} + \Gamma - \Lambda + \nabla \cdot (C_T \nabla T), \quad (2.3) \]

where \( \varepsilon \) is the internal energy density, \( P \) is the pressure, \( \Gamma \) is the local heating function, \( \Lambda \) is the local cooling function, \( C_T \) is the thermal conductivity, and \( T \) is the temperature.

Several secondary equations must still be provided to close the set of equations (2.1)–(2.3). If there is pressure, then there must be an equation of state. If there
is gravity, then there must be an equation defining the gravitational potential. If there is a magnetic field, then an equation specifying it must be given. If there are heating and cooling functions, then they must be specified.

To be specific, when \( f = -\nabla P - \rho \nabla \Phi \) and the gas is adiabatic, i.e., \( dQ \equiv \Gamma - \Lambda + \nabla \cdot (C_T \nabla T) = 0 \) where \( dQ \) is the heat transferred, then the energy equation (2.3) is unnecessary. In this case, the two secondary equations needed to close equations (2.1) and (2.2) are the barotropic equation of state \( P = K \rho^\gamma \) and Poisson’s equation \( \nabla^2 \Phi = 4\pi G \rho \), where \( K \) specifies the specific entropy, \( \gamma \) is the adiabatic exponent, \( \Phi \) is the gravitational potential, and \( G \) is the gravitational constant. When investigating a dynamical instability in rapidly rotating protostars, this is a good first approximation.

Before the hydrodynamic equations (2.1)-(2.3) are further manipulated, a few strategic decisions must be made concerning their solution. If the equations are going to be solved in their full three-dimensional generality, then they must be solved on a finite grid, since the continuum problem must be put on a finite computer. Only numerical solutions are possible in general.

Once an initial distribution of mass and fluid velocities are given, with whatever secondary variables are necessary for the set of equations to close, then the outcome is determined for all time. The solution at later times is quite often very sensitive to the initial conditions.

With these difficulties in mind, the first decision that must be made is whether to solve the partial differential equations using a “finite element” or a “finite difference” technique. Finite element analysis essentially involves expanding each
dynamical variable in terms of a complete set of functions and then using variational techniques to determine the unknown coefficients (Mitchell 1972). Since the systems I wish to describe are not stiff, finite element methods are probably inappropriate, so finite difference methods will be used instead. Another reason for using finite difference methods instead of finite element methods is purely historical (Griffiths 1986). Finite element analysis has its origin in structural engineering, and astrophysicists have just not talked to structural engineers very often. There is little experience in the astrophysical community in applying the finite element method. The mathematical theory of the finite element method is actually better developed than the finite difference method. Perhaps this mathematical connection has also inhibited the general implementation of the finite element method.

Now that the finite difference technique has been chosen, two more choices remain: whether to integrate the equations in time using an implicit or explicit method, and whether to structure the equations in Lagrangian or Eulerian coordinates. Roache (1976) gives an outstanding discussion of both choices and Tohline (1982) gives a brief review of the salient differences. A discussion of the choices I have made follows.

The next decision that must be made is whether the dynamical variables are to be updated implicitly or explicitly in time. Since equations (2.1)–(2.3) are first degree in time, they can be cast into the form:

\[ \frac{\partial g_i}{\partial t} + G_i(g) = 0 , \]

where \( g_i \) is some dynamical variable, and \( G_i(g) \) is generally some complicated
function of the dynamical variables and involves spatial gradients as well. One then approximates the time derivative by, for instance, letting

$$\frac{\partial g_i}{\partial t} \approx \frac{g_i(t + \delta t) - g_i(t)}{\delta t},$$  \tag{2.5}

where $\delta t$ is the time step. Substituting this into equation (2.4) and solving for the updated $g_i$ yields:

$$g_i(t + \delta t) \approx g_i(t) - \delta t G_i(g).$$  \tag{2.6}

The question then is, At what time are the dynamical variables inside the function $G_i(g)$ evaluated?

If the variables inside $G_i(g)$ are evaluated at the new time, then the method is implicit and equations (2.1)–(2.3), as well as all the secondary equations, must be solved simultaneously; this generally evolves inversion of large sparse matrices. In the implicit method there is no formal restriction on the size of the time step $\delta t$. Usually, $\delta t$ is chosen such that the exciting parts of the evolution are not missed. This method has obvious advantages if nothing very dynamic is expected to happen and is clearly the best way to evolve secular models, but particularly for multidimensional problems it is costly in computer memory and difficult to program. Modularity of subroutines is a real problem, and therefore trustworthiness of results suffers, since everything must be solved simultaneously for a single time step.

If the variables inside $G_i(g)$ are evaluated at the old time, the method is explicit and equations (2.1)–(2.3), as well as all the secondary equations, can
be solved serially. In the explicit method, the well-known Courant condition becomes the limit on the time step. The Courant condition is

$$\delta t < \min \left[ \frac{\delta x}{v(x)} \right] \quad \forall \, x \text{ in the model}, \quad (2.7)$$

where $\delta x$ is the displacement across a grid and $v(x)$ is the sum of all the relevant velocities in the $x$ grid cell. If $f$ in equation (2.2) contains a spatial gradient in the pressure (as it normally does for any gas cloud), then $v(x) = v_x + v_s + v_v$ where $v_x$ is the velocity in the $x$ directions, $v_s$ is the velocity of sound, and $v_v$ is the viscous velocity. My prescription for the time step, based on the Courant condition, in the computer code I wrote, will be explained in detail later (see §2.10).

The next choice is Lagrangian versus Eulerian coordinates. Lagrangian coordinates of dynamical variables trace unique fluid elements; total time derivatives represent the time rate of change of Lagrangian dynamical variables. The relationship between the partial derivative with respect to time and the total derivative with respect to time for any function $g(x; t)$ is (c.f., Landau and Lifschitz 1975)

$$\frac{dg(x; t)}{dt} = \frac{\partial g(x; t)}{\partial t} + \mathbf{v} \cdot \nabla g(x; t). \quad (2.8)$$

Since the grid moves with the fluid elements, it often becomes quite distorted in 3D; this makes expressions for spatial operators like $\nabla$ and $\nabla^2$ that appear in $f$ difficult to express. Most analytic solutions and many one-dimensional, numerical solutions of hydrodynamic flows are done in Lagrangian coordinates since, with total derivatives in time, analytic integration in time is conceivable
and occasionally possible. In explicit schemes, which are Courant limited, at least fluid velocities do not appear in the Courant limit in Lagrangian coordinates. This, though, is only an advantage when the flow is very supersonic so that $v(x)$ is controlled by the speed of sound and not by the supersonic flow velocities.

Eulerian coordinates of dynamical variables are fixed in space—or, at least they do not move precisely along with individual fluid elements—and partial time derivatives represent the time rate of change of Eulerian dynamical variables. Matter moves through the grid, and the grid boundaries are well determined beforehand, so the grid does not distort uncontrollably during the evolution. The spatial operators in $f$ are easy to express computationally. Analytic solutions in Eulerian coordinates are rare, but the human mind—at least this one—has an easier time thinking about dynamical variables like the mass density in Eulerian coordinates where mass flows through the grid. In explicit schemes, the Courant limit is smaller, since now $v(x)$ includes both the speed of sound in the grid and the fluid velocity.

There is a way to cheat on this Courant limit by using a moving Eulerian grid. In some sense, this produces a mixed Lagrangian and Eulerian formulation, since the best results are achieved when the coordinate grid moves approximately with the fluid. But, computationally, the Eulerian formulation is maintained even though the grid is moved. One then has to decide how to move the grid, and this changes the way the advection (see §2.3) is done; but it increases the Courant time, since it reduces the relevant velocities $v(x)$ within a grid cell $x$. Advection, of which more will be said in the following section when the hydrodynamic equations are put into conservative form, is more complicated
when employing a moving grid; and, unless there is very supersonic flow or a radial collapse calculation is being done, there is little advantage to using a moving grid in a second-order hydrocode. (In a first-order hydrocode there is some advantage in using a moving grid to reduce numerical viscosity. More will be said about this in the viscosity section of this chapter.)

All work herein is explicit Eulerian hydrodynamics.

2.2 Conservative Form

Now that the choice has been made to do “explicit finite difference, Eulerian 3-D hydrodynamics,” it proves useful to cast equations (2.1)–(2.3) in conservative form and to identify the fundamental dynamical variables. To make this more concrete and to prepare the way for discussing how the second-order accurate code was actually written, a specific choice for \( f \) in equation (2.1) is made:

\[
f = - \nabla P - \rho \nabla \Phi .
\]  

(2.9)

Putting this into equation (2.2) yields:

\[
\frac{d}{dt} \rho \frac{d\mathbf{v}}{dt} = - \nabla P - \rho \nabla \Phi .
\]  

(2.10)

Writing the principle equations in conservative form will guarantee that, in the absence of source terms, certain fundamental physical quantities will be appropriately conserved. The obvious advantage of this form will be seen in the next section when the divergence theorem is used. The equation of continuity, since it contains no sources, is the prototype for the solution of the other equations.
Using equation (2.8) in equation (2.1) yields:

\[ \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0, \quad (2.11) \]

the conservative form of the continuity equation. The mass density \( \rho \) is the fundamental dynamical variable in this expression, indicating that mass is conserved.

Using equation (2.8) in equation (2.10) in Cartesian coordinates yields:

\[ \frac{\partial \mathbf{M}}{\partial t} + \nabla \cdot (\mathbf{M} \mathbf{v}) = -\nabla P - \rho \nabla \Phi, \quad (2.12) \]

where \( \mathbf{M} = \rho \mathbf{v} \) is the Cartesian momentum density, and \( \mathbf{v} \equiv (v_x, v_y, v_z) \equiv (M_x/\rho, M_y/\rho, M_z/\rho) \). The fundamental dynamical variables in equation (2.12) are clearly \( M_x, M_y, \) and \( M_z \), and \( \mathbf{v} \) has become a slave to \( \mathbf{M} \) and \( \rho \). Equations (2.11) and (2.12) are nonlinear, coupled partial differential equations.

The energy equation should also be cast into the conservative form. Using equation (2.8) in equation (2.3) yields:

\[ \frac{\partial \varepsilon}{\partial t} + \nabla \cdot (\varepsilon \mathbf{v}) = -P \nabla \cdot \mathbf{v} + \Gamma - \Lambda + \nabla \cdot (C_T \nabla T). \quad (2.13) \]

If \( P \) is given by a unique function of \( \rho \), such as through a barotropic equation of state, then the energy equation is not needed. The energy equation must be solved when there are heating and cooling terms and/or thermal heat conduction, or when there are shocks, or when the gas is not homentropic. If the pressure of the gas is ideal, then

\[ P = (\gamma - 1)\varepsilon. \quad (2.14) \]

Using this relation, the \(-P \nabla \cdot \mathbf{v}\) term in equation (2.13) can be expressed in
terms of $\varepsilon$ and, hence, combined with the terms on the LHS of (2.13) to give

$$\frac{d\varepsilon}{dt} + \gamma \varepsilon \nabla \cdot \mathbf{v} = \Gamma - \Lambda + \nabla \cdot (C_T \nabla T). \quad (2.15)$$

This equation is distinctly not in conservative form, indicating that the internal energy itself is not the quantity likely to be conserved in a hydrodynamic flow. Tohline (1988) has pointed out, however, that equation (2.15) can be cast in the form

$$\frac{\partial \varepsilon^{1/\gamma}}{\partial t} + \nabla \cdot (\varepsilon^{1/\gamma} \mathbf{v}) = \frac{1}{\gamma} \varepsilon^{\frac{\gamma}{\gamma-1}} - \frac{1}{\gamma} \left[ \Gamma - \Lambda + \nabla \cdot (C_T \nabla T) \right], \quad (2.16)$$

which is conservative in terms of the dynamical variable $\varepsilon^{1/\gamma}$, a tracer of the entropy. Entropy is conserved when $dQ = 0$, and then the RHS of equation (2.16) is zero. The advantage in expressing the $-P \nabla \cdot \mathbf{v}$ term from (2.12) as part of the advection term instead of as a source term will become apparent latter.

The reason equations (2.1)--(2.3) were transformed into the forms given by equations (2.11), (2.12), and (2.16) is so that the divergence terms (alternatively referred to as "advection" or "flux" terms) can be separated from the source terms. When integrating this coupled set of equations numerically, the flux terms and source terms will be handled quite differently. In fact, in the hydrocode they will even be done in separate subroutines. Handling these terms in separate subroutines is more than just a programming convenience to assure modularity of the code. It helps to ensure a physically realistic solution to the mathematical equations; numerical experiments have demonstrated this (see §2.4).

Since both the first-order hydrocode (written by Tohline 1978, 1980, 1982) and the second-order hydrocode (written by me) use a polar cylindrical coor-
dinate system, equation (2.12) needs to be written in cylindrical coordinates. Here, polar cylindrical coordinates have their usual meaning, but to ensure no misunderstanding, the coordinate transformations from cylindrical to Cartesian coordinates are:

\[(r, z, \theta) = (\sqrt{x^2 + y^2}, z, \arctan(y/x))\].  

(2.17)

The continuity equation and the energy equation are scalar equations for advancing \( \rho \) and \( \varepsilon \) (or \( \varepsilon^{1/\gamma} \)) in time, so their specification into a cylindrical coordinate system is trivial, involving only putting the gradients and divergences in the proper coordinate system. The equation of motion is a vector equation, and simply replacing the gradient and divergence operators with their proper expressions in cylindrical coordinates will not work because the momentum densities in equation (2.12) are in Cartesian, not cylindrical, coordinates. The direction of the cylindrical \( \hat{r} \) unit vector is not parallel for different values of \( \theta \) in the coordinate manifold, so the affine, which is used in parallel transport, is nonzero in this curvilinear coordinate system. Another way of putting this is that there is a centripetal acceleration term in the radial momentum density because of the coordinate system. Within a cylindrical coordinate system, after some algebra the three components of the modified version of equation (2.12) become:

\[
\frac{\partial S}{\partial t} + \nabla \cdot (S\mathbf{v}) = -\rho \frac{\partial \Phi}{\partial r} - \frac{\partial P}{\partial r} + \frac{A^2}{\rho r^3},
\]

(2.18)

\[
\frac{\partial T}{\partial t} + \nabla \cdot (T\mathbf{v}) = -\rho \frac{\partial \Phi}{\partial z} - \frac{\partial P}{\partial z},
\]

(2.19)

\[
\frac{\partial A}{\partial t} + \nabla \cdot (A\mathbf{v}) = -\rho \frac{\partial \Phi}{\partial \theta} - \frac{\partial P}{\partial \theta},
\]

(2.20)
where $S$ is the radial momentum density $S = \rho v_r$, $T$ is the $z$-direction momentum density $T = \rho v_z$, and $A$ is the angular momentum density $A = r \rho v_\theta$. The velocities are slaves of the momentum densities, so

$$v \equiv (v_r, v_z, v_\theta) \equiv (S/\rho, T/\rho, A/(r \rho)) .$$  \hspace{1cm} (2.21)

### 2.3 Sourcing and Fluxing

All five of the scalar equations (2.11), (2.18)-(2.20), and (2.16) can be cast into the general form:

$$\frac{\partial B}{\partial t} + \nabla \cdot (Bv) = S_B ,$$  \hspace{1cm} (2.22)

where $S_B$ is the source term for the dynamical variable $B$. The source term for the equation of continuity is

$$S_\rho \equiv 0 .$$  \hspace{1cm} (2.23)

The source terms for the equation of motion are:

$$S_S = -\rho \frac{\partial \Phi}{\partial r} - \frac{\partial P}{\partial r} + \frac{A^2}{\rho r^3} ,$$  \hspace{1cm} (2.24)

$$S_T = -\rho \frac{\partial \Phi}{\partial z} - \frac{\partial P}{\partial z} ,$$  \hspace{1cm} (2.25)

$$S_A = -\rho \frac{\partial \Phi}{\partial \theta} - \frac{\partial P}{\partial \theta} .$$  \hspace{1cm} (2.26)

The source term for the energy equation is:

$$S_\varepsilon = \Gamma - \Lambda + \nabla \cdot (C_T \nabla T) .$$  \hspace{1cm} (2.27)
Equation (2.22) can be cast into integral form:

$$\int \frac{\partial B}{\partial t} d\tau + \int \nabla \cdot (Bv) d\tau = \int S_B d\tau , \tag{2.28}$$

where $\tau$ is the volume of a grid cell anywhere. Commuting the partial derivative past the integral sign in the first term of the above equation and applying the divergence theorem to the second term yields:

$$\frac{\partial}{\partial t} \int B d\tau + \int Bv \cdot dA = \int S_B d\tau , \tag{2.29}$$

where $dA$ is the infinitesimal surface vector. If the grid is not moving, then $\tau$ is stationary and the volume integrals are done. [If the grid is allowed to move in such a way that expansion or contraction of individual grid cells can occur, then $\tau(t) \neq \tau(t + \delta t)$ in general, and the integral cannot be directly commuted with the partial derivative with respect to time. See Norman and Winkler (1986) for an example of how the effect of a moving grid can be incorporated.] Now, applying a simple time difference to evaluate the time derivative and writing the integral over the surface of the volume $\tau$ as $[\ ]|_A$ yields:

$$\frac{B(t + \delta t) - B(t)}{\delta t} \tau + [B^* v \cdot \delta A]|_A = S_B \tau , \tag{2.30}$$

where $\delta A$ is the finite surface vector. Putting an asterisk on $B$ within the surface integral means that this is $B$ at the surface of the volume $\tau$, where $B$ without an asterisk denotes the variable value in the center of the volume $\tau$. Solving for the
updated variable $B$ yields:

$$B(t + \delta t) = B(t) - \frac{\delta t}{\tau} [B^* \cdot \delta A]_A + S_B \delta t. \quad (2.31)$$

Notice that the velocities used in fluxing are the velocities at the surface of the volume $\tau$, the so-called control volume.

The term $\frac{\delta t}{\tau} [B^* \cdot \delta A]_A$ is the advection term. This term is caused by fluid flowing through the grid. When fluxing $S$, $T$, or $A$, while solving the equation of motion, this term is obviously nonlinear. The advection term is what makes hydrodynamics so difficult to handle numerically as well as analytically. A more accurate advection term improves the hydrodynamic simulation. The source terms are relatively trivial to handle numerically, with spatial derivatives being approximated by finite differences in fairly obvious ways.

Exactly how the advection is done in my cylindrical coordinate treatment will now be demonstrated. Figure 2.1 makes some of the geometry clear. The integral over the surface in cylindrical coordinates is represented simply by the sum over the six sides of the pie-shaped volume $\tau$: 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 2.1.—A typical pie-shaped volume $\tau$ in cylindrical coordinates, showing the grid lines.
\[ [B^*(t + \frac{1}{2} \delta t) \cdot \delta A]_A = \]

\[ B^*(r + \delta r) \cdot v_r(r + \delta r) \cdot \delta z \cdot \delta \theta \cdot [r + \delta r - v_r(r + \delta r) \cdot \frac{1}{2} \delta t] \]

\[ - B^*(r) \cdot v_r(r) \cdot \delta z \cdot \delta \theta \cdot [r - v_r(r) \cdot \frac{1}{2} \delta t] + \]

\[ B^*(z + \delta z) \cdot v_z(z + \delta z) \cdot \frac{1}{2} \delta \theta \cdot \{[r + \delta r]^2 - r^2\} \]  \hspace{1cm} (2.32)

\[ - B^*(z) \cdot v_z(z) \cdot \frac{1}{2} \delta \theta \cdot \{[r + \delta r]^2 - r^2\} + \]

\[ B^*(\theta + \delta \theta) \cdot v_\theta(\theta + \delta \theta) \cdot \delta z \cdot \delta r \]

\[ - B^*(\theta) \cdot v_\theta(\theta) \cdot \delta z \cdot \delta r . \]

In order to reduce cluttering in (2.32), the quantities \( B \) and \( (v_r, v_z, v_\theta) \) are specified at time \( t \) and at the grid location \((r, z, \theta)\), unless otherwise noted. So, for example, \( B^*(z + \delta z) \equiv B^*(r, z + \delta z, \theta) \). The six terms in equation (2.32) are the \( B \) fluxes, and henceforth will be denoted symbolically as:

\[ [B^*(t + \frac{1}{2} \delta t) \cdot \delta A]_A = \]

\[ F^B_r(r + \delta r, z, \theta) \]

\[ - F^B_r(r, z, \theta) + \]

\[ F^B_z(r, z + \delta z, \theta) \]  \hspace{1cm} (2.33)

\[ - F^B_z(r, z, \theta) + \]

\[ F^B_\theta(r, z, \theta + \delta \theta) \]

\[ - F^B_\theta(r, z, \theta) . \]

A second-order accurate interpolation scheme is differentiated from a first-order accurate scheme primarily by how \( B^* \) is calculated. When using a cylindrical coordinate grid, however, there will also generally be a difference in how the areas of the two radially directed cell faces are calculated in the flux terms.
Equation (2.32) is set up for a second-order scheme; in a first-order scheme, the surface area \( \delta z \delta \theta [r + \delta r - v_r(r + \delta r) \frac{1}{2} \delta t] \) is replaced by \( \delta z \delta \theta [r + \delta r] \) and the area \( \delta z \delta \theta [r - v_r(r) \frac{1}{2} \delta t] \) is replaced by \( \delta z \delta \theta r \).

In discrete hydrodynamics one constantly needs values of functions at spatial and temporal locations other than where they are normally tabulated: thus the need for interpolation with, in principle, the highest degree of accuracy possible. The problem is that numerical schemes employing high-order accuracy are often unstable and lead to saw-tooth oscillations of the dynamical variables in every other zone. In multidimensional flows, we have even found that a numerical instability originating in only one direction can propagate into the other dimensions as well. This is why developing a scheme that is both of high order and stable is so important.

If \( S, T, A, \) and \( \varepsilon \) are centered at the same spatial location that \( \rho \) is, then the velocities at the center of the faces of the volume \( r \) that are needed to do the fluxes are

\[
v_r(r, z, \theta) = \frac{1}{2} \left[ T(r, z - \delta z, \theta) + T(r, z, \theta) \right] / \rho(r, z, \theta),
\]

\[
v_z(r, z, \theta) = \frac{1}{2} \left[ S(r - \delta r, z, \theta) + S(r, z, \theta) \right] / \rho(r, z, \theta),
\]

\[
v_\theta(r, z, \theta) = \frac{1}{2(r + \frac{1}{2} \delta r)} \left[ A(r, z, \theta - \delta \theta) + A(r, z, \theta) \right] / \rho(r, z, \theta).
\]

Given a discrete time step \( \delta t \), the general procedure that is used to advance any dynamical variable \( B \) in time is to calculate the values of both the "source" term and the "flux" term in relation (2.22), then, as prescribed by that equation, add or subtract them from the current value of \( B \).
2.4 SOURCING AND FLUXING SEPARATELY

In practice, what I currently do is split the sourcing and fluxing operations. First, the dynamical equations are "sourced," updating the dynamical variables by accelerations in the equation of motion and by nonadiabatic sources in the energy equation. The velocities are then updated throughout the computational grid by using definition (2.21) and the new values of $S$, $T$, $A$, and $\rho$. The equations are then "fluxed" by adverting material through the faces of each computational grid cell. Secondary equations are solved to update $P$ and $\Phi$. Finally, the new time step is calculated using the Courant time-step limit. This procedure is repeated for each successive time step.

Norman and Winkler (1986) point to the importance of handling the advection terms separately from the source terms. They mention that energy is conserved more accurately if it is done that way. My own numerical experiments have convinced me that fluxing and sourcing separately, with a velocity update in between, is not something that is just nice, and therefore optional, but is fairly essential, especially in models that are not extremely dynamic.

In order to emphasize this point, let me recount a specific incident that happened while I was testing my newly developed, second-order scheme. In a very slowly rotating, equilibrium polytrope, for example, as generated by Hachisu as an input to my hydrocode, pressure is balanced by gravity and the model is essentially spherical with density decreasing monotonically with radius. This is a dull model and its evolution should stay dull. For example, isodensity contours are initially spheres and they should stay spheres. If anything happens in a
model like this in a hydrodynamic code, something is wrong with the code. (In my opinion this model should be one of the test cases run by anyone writing a new hydrocode, and it should be run on all existing codes. A code should pass this test.) Upon hearing that Tohline was seeing a substantial amount of "noise" develop spontaneously in rapidly rotating models evolved with his first-order code, Durisen examined the behavior of a nonrotating equilibrium model using Tohline's code. Durisen saw structures develop in the azimuthal direction and told us about his results. Durisen's experiment hinted strongly that the "noise" Tohline had been seeing was numerical rather than physical. (This is always a problem in numerical hydrodynamics, one that hydrodynamicists worry about, but often it is a problem discussed only between themselves and quite often in whispers. After all, writing several thousand lines of code and getting it to work requires an enormous investment of soul and ego.) When Tohline modified his code to perform sourcing and fluxing separately on the \( A \) component of the equation of motion, the problem of azimuthal structures in the nonrotating model went away. When I did the same in my second-order code, azimuthal "noise" disappeared as well. I then graphed everything in the model that I could think of, as I am prone to do for any model, and saw that, while structures were no longer developing azimuthally, structures were still developing in the \( r-z \) contour plots of the density. With \( S \) and \( T \) sourced and fluxed separately, the structures disappeared in the slowly rotating model in \( r \) and \( z \), too.

It was then necessary to determine if the large-scale dynamical features—ordered structures that had generally been larger in amplitude than the "noise"—we had obtained before separating the sourcing and fluxing operations were valid.
Several previously run dynamical models were then run in both Tohline's and my code with sourcing and fluxing done separately. The general results that had been seen before appeared again. The procedure of sourcing and fluxing separately, with a velocity update in between, in no way averages out real physical structures on the grid, only numerical ones. If the model is rapidly evolving it seems not to matter if you source and flux separately, since the dynamic evolution seems to wash out the numerical noise induced by not sourcing and fluxing separately. Since the velocity update costs so little in terms of computer programming and running time, we think that it should always be done.

The procedure, then, is to first source the dynamical variables $S$, $T$, $A$, and $\varepsilon$ ($\rho$ is not sourced because it has no source terms). Therefore, all the accelerations of $S$, $T$, and $A$, and the nonadiabatic (irreversible) heating, cooling, and heat conduction of $\varepsilon$ are done in the source step. The velocities $v_r$, $v_z$, and $v_\theta$ are updated. Then these new velocities are used to adjust $\rho$, $S$, $T$, $A$, and $\varepsilon^{1/\gamma}$ in the fluxing step.

### 2.5 First-Order Fluxing, Donor Cell

As was pointed out in §2.3, when you flux a dynamical variable using equation (2.32), how $B^*$ is calculated depends upon what order fluxing scheme is used. In first-order, "donor cell" fluxing, $B^*$ is just the density of the fluid at the center of the grid cell from which the flow is coming. So for the $r$ face of the pie-shaped volume $r$,

$$B^*(r) \equiv \begin{cases} B(r - \delta r) & v_r(r) \geq 0, \\ B(r) & v_r(r) < 0. \end{cases}$$

(2.37)
Similarly, for the $z$ face,

\[
B^*(z) = \begin{cases} 
B(z - \delta z) & v_z(z) \geq 0, \\
B(z) & v_z(z) < 0.
\end{cases}
\]  
(2.38)

Then for the $\theta$ face,

\[
B^*(\theta) = \begin{cases} 
B(\theta - \delta \theta) & v_\theta(\theta) \geq 0, \\
B(\theta) & v_\theta(\theta) < 0.
\end{cases}
\]  
(2.39)

### 2.6 Second-Order Fluxing, Van Leer Monotonic Interpolation

In second-order fluxing, the general idea is to determine the value of $B^*$ at, or at least near to, the face of the cell at which the flow is occurring by linearly interpolating between the cell-center values on either side of the face. It is not good to take a strict average of the two densities because this scheme is not stable and will lead to the saw-tooth instability mentioned above. Van Leer (1977 a, 1977 b, and 1979) invented a monotonic interpolation scheme that prevents this oscillation, and is, in principle, of any order. Van Albada, van Leer, and Roberts (1982) did a comparative study of this and other algorithms as computational methods in cosmic gas dynamics. Norman and Winkler (1986) clearly describe the implementation of the second-order van Leer monotonic interpolation scheme in their 2-D Eulerian hydrodynamic code.

To understand second-order van Leer monotonic interpolation, consider first a problem in one dimension. (The generalization to 3D in cylindrical coordinates is given in the next paragraph.) Consider a tabulated function $B$ known at \{$B_i$\} on a manifold \{$x_i$\}. A second-order interpolating function, $B_i(\lambda) = B_i + \Delta B_i \lambda$
where $-\frac{1}{2} \leq \lambda \leq \frac{1}{2}$ and $\lambda$ is the normalized distance from the zone's center, will result if $B$ is a piecewise linear distribution. The quantity $B_i$ will be a zone average since $B_i = \int B_i(\lambda) \, d\lambda$. The van Leer (1977b) monotonic interpolation scheme chooses the largest $|\Delta B_i|$ such that $B_i(-\frac{1}{2})$ and $B_i(\frac{1}{2})$ do not exceed the neighboring zone averages $B_{i-1}$ and $B_{i+1}$, and if $B_i$ is a local extremum then $\Delta B_i = 0$. The wondrous $\Delta B_i$, the van Leer slope, that satisfies these conditions is

$$
\Delta B_i \equiv \begin{cases} 
2\delta B_i\delta B_{i+1}/[\delta B_i + \delta B_{i+1}] & \delta B_i\delta B_{i+1} > 0, \\
0, & \delta B_i\delta B_{i+1} \leq 0,
\end{cases} \tag{2.40}
$$

where $\delta B_i \equiv B_i - B_{i-1}$. Notice that the van Leer slope $\Delta B_i$ depends only on $B_{i-1}$, $B_i$, and $B_{i+1}$. The normalized distance from the zone's center $\lambda$ that is used to compute the flux $F_i = B^*_i v_i \text{Area}_i$ through the $i^{th}$ face will be such that $B^*_i$ is

$$
B^*_i \equiv \begin{cases} 
B_{i-1} + \left\{ \frac{1}{2} - v_i\frac{3}{2}\delta t/\delta x_i \right\} \Delta B_{i-1} , & v_i \geq 0, \\
B_i - \left\{ \frac{1}{2} + v_i\frac{3}{2}\delta t/\delta x_i \right\} \Delta B_i , & v_i < 0, 
\end{cases} \tag{2.41}
$$

where $\delta x_i \equiv x_{i+1} - x_i$. [In Norman and Winkler (1986), there is a typographical error in their equation (36), which is corrected in equation (2.41) here. Equation (36) in Norman and Winkler (1986) is inconsistent both with their own derivation and with the correct equation (5.14) published in Winkler and Norman (1986a).] (Again, in 3D, using cylindrical coordinates only $\text{Area}_r$, the radial faces area, will be different in first- and second-order interpolation. This is because in equation (2.32) $\text{Area}_r$ is a function of $r$, $\text{Area}_z$ is not a function of $z$, and $\text{Area}_\theta$ is not a function of $\theta$.) Notice that if $B_i$ is a local extremum, fluxing in second order reduces to the first-order approximation; this is what makes the scheme stable.
The following is an exact description of what I have implemented in my 3-D code for the fluxing of $B$, which is any dynamical variable. So, for the $r$ face of the pie-shaped volume $r$:

$$B^*(r) \equiv \begin{cases} 
B(r - \delta r) + \left\{ \frac{1}{2} - v_r(r) \frac{1}{2} \frac{\delta t}{\delta r} \right\} \Delta B(r - \delta r) & v_r(r) \geq 0, \\
B(r) - \left\{ \frac{1}{2} + v_r(r) \frac{1}{2} \frac{\delta t}{\delta r} \right\} \Delta B(r) & v_r(r) < 0,
\end{cases} \tag{2.42}$$

where $\Delta B(r)$ by van Leer monotonic interpolation is:

$$\Delta B(r) \equiv \begin{cases} 
2\delta B(\delta B(r + \delta r) / [\delta B(r) + \delta B(r + \delta r)] & \delta B(r) \delta B(r + \delta r) > 0, \\
0 & \delta B(r) \delta B(r + \delta r) \leq 0,
\end{cases} \tag{2.43}$$

and $\delta B(r) \equiv B(r) - B(r - \delta r)$.

Similarly for the $z$ face:

$$B^*(z) \equiv \begin{cases} 
B(z - \delta z) + \left\{ \frac{1}{2} - v_z(z) \frac{1}{2} \frac{\delta t}{\delta z} \right\} \Delta B(z - \delta z) & v_z(z) \geq 0, \\
B(z) - \left\{ \frac{1}{2} + v_z(z) \frac{1}{2} \frac{\delta t}{\delta z} \right\} \Delta B(z) & v_z(z) < 0,
\end{cases} \tag{2.44}$$

$$\Delta B(z) \equiv \begin{cases} 
2\delta B(z) \frac{\delta B(z + \delta z)}{[\delta B(z) + \delta B(z + \delta z)]} & \delta B(z) \delta B(z + \delta z) > 0, \\
0 & \delta B(z) \delta B(z + \delta z) \leq 0,
\end{cases} \tag{2.45}$$

and $\delta B(z) \equiv B(z) - B(z - \delta z)$.

Then for the $\theta$ face:

$$B^*(\theta) \equiv \begin{cases} 
B(\theta - \delta \theta) + \left\{ \frac{1}{2} - v_\theta(\theta) \frac{1}{2} \frac{\delta t}{\delta \theta} \right\} \Delta B(\theta - \delta \theta) & v_\theta(\theta) \geq 0, \\
B(\theta) - \left\{ \frac{1}{2} + v_\theta(\theta) \frac{1}{2} \frac{\delta t}{\delta \theta} \right\} \Delta B(\theta) & v_\theta(\theta) < 0,
\end{cases} \tag{2.46}$$

$$\Delta B(\theta) \equiv \begin{cases} 
2\delta B(\theta) \frac{\delta B(\theta + \delta \theta)}{[\delta B(\theta) + \delta B(\theta + \delta \theta)]} & \delta B(\theta) \delta B(\theta + \delta \theta) > 0, \\
0 & \delta B(\theta) \delta B(\theta + \delta \theta) \leq 0,
\end{cases} \tag{2.47}$$

and $\delta B(\theta) \equiv B(\theta) - B(\theta - \delta \theta)$. Notice if all the $\Delta$ terms, which are the van
Leer monotonic second-order interpolating slopes, are set equal to zero, then first-order fluxing is recovered.

2.7 Consistent Advection

In each run so far, each dynamical variable $B$ has been fluxed independently. In contrast, in consistent advection everything is advected with the mass density. Consistent advection is an idea that can be discussed and developed independently of one's choice of a particular interpolation scheme—first order, second order, $n^{th}$ order. The specific implementation of consistent advection will, however, depend upon which interpolation is being used.

Before implementing van Leer monotonic interpolation, Norman, Wilson, and Barton (1980) recommended using consistent advection of angular momentum, and they showed that it improved local conservation of angular momentum. They advected the specific angular momentum $J = A/\rho$ with the mass flux $F\rho$, using an angular momentum flux

$$F^A = J^*F\rho .$$

In the absence of viscous torques, the specific angular momentum is conserved along fluid stream lines in an axially symmetric flow, so the argument goes that spatial interpolation should be on $J$ rather than on $A$. In a 3-D code there is no axial symmetry artificially enforced, but while the physical problem being investigated remains axisymmetric the code should not introduce spurious nonaxisymmetric terms.
In a rapidly rotating, equilibrium polytrope, $\rho$ and $\Omega$ are fairly flat (constant in $r$) near the rotation axis. Let's examine how a consistent advection scheme would fare in a model of this sort. Since $A = \rho \Omega r^2$, if you have uniform rotation ($\Omega =$constant) and uniform density ($\rho =$constant), $A$ would not be approximated well by the usual fluxing methods, which either assume that $A$ is constant within a cell (first order) or linearly interpolated to a value at a cell face (second order). It is obvious that, for small $r$ (i.e., near the rotation axis), and $\Omega$ and $\rho$ constant, the pedestrian way of advecting just as $\rho$ was advected would lead to gross inaccuracies, particularly in the fluxes through the $r$ face. In the first two or three central zones of our simulations some non-physical garbage is always seen. I would like to eliminate it fairly, so a more elaborate scheme should be used.

Norman and Winkler (1986) recommend an even more complicated way of advecting $A$ through the radial faces of each grid cell, using van Leer monotonic interpolation on only the flattest of three angular quantities. Specifically,

$$F^A_r = J^{**} F^\rho_r,$$

(2.49)

where

$$J^{**} = \begin{cases} 
\Omega^*[r - v_r \frac{1}{2} \delta t]^2, & \text{if } |\Delta \Omega/\Omega| \text{ smallest,} \\
W^*[r - v_r \frac{1}{2} \delta t], & \text{if } |\Delta W/W| \text{ smallest,} \\
J^*, & \text{if } |\Delta J/J| \text{ smallest,}
\end{cases}$$

and the two new angular quantities are defined as

$$W \equiv J \frac{3(r + \frac{1}{2} \delta r)}{(r + \delta r)^2 + r(r + \delta r) + r^2},$$

(2.50)

$$\Omega \equiv J \frac{2}{(r + \delta r)^2 + r^2}.$$

(2.51)

In these various expressions, a single asterisk means to do a van Leer monotonic

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
interpolation, and \( \Delta \) means to do a van Leer slope.

Advection of \( A \) on the \( z \) and \( \theta \) faces is much simpler and is just

\[
F_z^A = J_z^* F_\rho^\rho ,
\]

(2.52)

\[
F_\theta^A = J_\theta^* F_\rho^\rho .
\]

(2.53)

Consistent advection with the mass, during the flux stage for \( S \) and \( T \), can be accomplished by simply using the mass fluxes calculated during the fluxing of \( \rho \) and multiplying them by the velocities at the faces of the pie-shaped volume. The fluxes in \( S \) and \( T \) become:

\[
F^S = v_r F^\rho ,
\]

(2.54)

\[
F^T = v_\theta F^\rho .
\]

(2.55)

In a system where \( \rho \), \( S \), \( T \), \( A \), and \( \varepsilon \) are all calculated at the same zone center (which is what I currently do) and therefore share the same control volume, one does not need to interpolate on the velocities of the faces other than to use equations (2.34)–(2.36). If \( S \) and \( T \) are centered elsewhere, then an additional interpolation will have to be done. In this case, if the interpolation is van Leer monotonic second order then one will have to interpolate when calculating the mass fluxes as spelled out in §2.6 and then interpolate again on the velocities in a similar manner. Norman and Winkler (1986) center \( S \) and \( T \) on faces of \( r_\rho \), the control volume for \( \rho \), and therefore must do a van Leer interpolation on \( v_r \).
and $v_z$ in equation (2.54) and (2.55). This is why their equations (54) and (55) have an asterisk on the velocities.

Consistent advection has not yet been incorporated in my second-order code, but it is only a matter of time until I try various combinations to see which choice works out the best. Consistent advection will be somewhat easier to compute when $S$, $T$, $A$, and $\rho$ are centered in the same place and share the same control volume. Having only one centering also requires having only one control volume $r$, and one does not have to distinguish different control volumes with dynamical variables centered elsewhere, nor does one need to have multiple ways of calculating the face-centered velocities on the different control volumes. This saves in code writing, in code verification, and in computer run time, since the same face-centered velocities may be used. In any 3-D code one has to be very careful what goes into the triply nested do loops. One does not want to calculate the same quantity many times. Much more will be said about this in the next chapter, “The Computer Code Itself.”

2.8 Errors Caused by Interpolation

Bickley (1948) derived error terms for finite difference formulae for a square lattice. His method is generally applicable to finite difference formulae on any type of lattice. The main idea is to write the displacement operator, or symbolic Taylor’s theorem, as an exponential:

$$f(x + \delta x) = e^{\delta x \partial_x} [f(x)].$$

(2.56)

It is then straightforward to derive error terms in the displacement $\delta x$ for interpo-
lated values on a grid, or to derive error terms in the displacement for differential operators approximated by a finite difference. When this is done, if the leading term in the error is proportional to $\delta x$, then the expression is first-order accurate in the grid spacing; if the leading term is proportional to $(\delta x)^2$, then the expression is second-order accurate in the grid spacing. Equation (2.56) is substituted into the interpolation and the exponential operator is expanded. The interpolation is then algebraically manipulated until it is in the form:

$$\text{interpolated definition} = \text{true expression} + \text{error terms}.$$  \hfill (2.57)

The important results that (a) centered differences for partial derivatives and centered averages on a uniform grid are second-order accurate and that (b) the donor cell methods are first-order accurate can be demonstrated by using this method. The first-order nature of the donor cell advection scheme is particularly easy to demonstrate: doing so, for example, does not exceed my personal mistake frequency. When one wants to derive error terms for more complicated expressions, however, it is nice to have access to algebraic manipulation programs like SMP or MACSYMA, or, if nothing else is available, REDUCE. (I have acquired a manual on IBM's successor to FORMAC called SCRATCHPAD, but I have never had an account on a computer with SCRATCHPAD. The manual makes SCRATCHPAD look capable of doing sophisticated algebra.)

Given that a method is second-order accurate, for instance, it is important to realize that not all second-order methods have the same factor multiplying the $(\delta x)^2$ term. One of the reasons I chose to center everything at the same place on the grid in my second-order code is that the $A^2/(r^3 \rho)$ term in the sourcing
of $S$ has zero error with this centering, but nonzero error with other standardly chosen centering.

2.9 Viscosity

There are four different types of viscosity typically encountered in astrophysical hydrodynamical simulations: molecular, turbulent, numerical, and artificial. In the equation of motion, molecular viscosity adds the terms

$$\frac{1}{3} \nabla (\nabla \cdot \mathbf{v}) + \nu \nabla^2 \mathbf{v} \quad (2.58)$$

to $f$, where $\nu$ is the coefficient of kinematic viscosity; and in the energy equation it adds the source term

$$\nu \nabla \cdot \mathbf{v} \quad (2.59)$$

Modeling turbulence as viscosity is an ad hoc technique whereby a viscosity is put in as if it were molecular, i.e., the form of terms in the equation of motion and in the energy equation are identical to a molecular (real) viscosity—but the molecular viscosity is actually much smaller. The rationale is that there are large-scale random motions in the fluid (larger than the mean free path of the gas molecules), but the scale of the motion is small compared to the size of the physical system being modeled. This causes the flow velocities to change, and viscosity is one way to change velocities. The kinematic viscosity is $\nu = \nu_t l$, where $\nu_t$ is the turbulent velocity, and $l$ is the characteristic length scale over which that section of the fluid maintains a turbulent velocity $\nu_t$. If $\nu_t$ is large then $l$ can be small, and if $l$ is large then $\nu_t$ can be small; however $l$ must be
smaller than the grid size. One just takes on faith that the product of \( vl \) is the same for all \( v_t \) and \( l \); and that \( \nu \) can be chosen for convenience' sake. The Doppler velocity dispersion of a gas might give you an estimate of \( v_t \) when the spectral line width from a gas is not determined by temperature. The use of turbulent viscosity in a simulation is a phenomenological solution to a problem that cannot be addressed directly or simply. Like all phenomenological solutions, if you are careful, it is hoped that the character of the solution is preserved even if the details are left out. A thought-provoking article on subgrid-scale flow with adequate reference for future study is Marcus (1986). Turbulent viscosity does not appear in the hydrocode that I have written, though it could be added easily enough as an additional source term.

So-called numerical viscosity is introduced into a model by explicit time integration using a finite difference technique. In a first-order code this can be appreciable (Norman, Wilson, and Barton 1980). In some cases, it is possible to correct for the effect of numerical viscosity on the growth rate of nonaxisymmetric structures; see Tohline, Durisen, and McCollough (1985) and Williams and Tohline (1987 a, also reprinted in Appendix A). Numerical diffusion does not affect the pattern speed of nonaxisymmetric structures greatly except through higher-order terms in first-order codes. The biggest motivation for writing a second-order accurate, finite difference code is that numerical viscosity is practically nonexistent in such a code.

When shocks develop, artificial viscosity can be used to mediate the shocks so they do not spread, but stay as one or two zones. The shock width in astrophysics is usually much too small to be resolved on a reasonably sized computational grid.
The method of using artificial viscosity has been used since antiquity (well, at
least since the early development of electronic computers); see Roache (1976). In
Tohline’s first-order code, no artificial viscosity is needed because the numerical
viscosity itself is large enough to mediate shocks well, at least in the problems that
have been explored with that code, to date. In my second-order code, artificial
viscous stress is used to mediate the numerical shock transitions, because there is
very little diffusive numerical viscosity in a second-order method. Another way
to handle this shock problem would be to handle the jump conditions exactly.
Handling the jump conditions at a shock is very computer intensive and has been
done by Woodward and collaborators; see Woodward (1986), Fryxell, Woodward,
Colella, and Winkler (1986), Woodward and Colella (1984), and Colella and
Woodward (1984). I do not currently use a shock-fitting technique because 3-D
codes are already computer intensive.

Norman and Winkler (1986), in their 2-D explicit Eulerian code, recommend
using a quadratic artificial viscosity stress tensor that is diagonal, \( Q \), of the form:

\[
Q_{ij} \equiv C_2 \rho \delta_{ij} \delta v_i \min[\delta v_i, 0] \quad i = r, z, \theta ,
\]

(2.60)

where \( C_2 \) is a constant of order unity (it is unity in my code), \( \delta_{ij} \) is the Kronecker
delta, and \( \delta v_i \) is the difference in velocity across the cell in the \( i^{th} \) direction. This
prescription for artificial viscosity will be sensitive to the compression of the
shock fronts and will give the width of numerical shock fronts the same thickness
regardless of where they are in the coordinate grid. Introducing artificial viscosity
will affect the equation of motion (2.10) by adding another source term, \( -\nabla \cdot Q \),
so the equation of motion becomes:

$$\frac{dv}{dt} \rho = -\nabla P - \rho \nabla \Phi - \nabla \cdot Q . \quad (2.61)$$

In cylindrical coordinates the conservative form of equation of motion, with artificial viscosity included, becomes:

$$\frac{\partial S}{\partial t} + \nabla \cdot (Sv) = -\rho \frac{\partial \Phi}{\partial r} - \frac{\partial P}{\partial r} + \frac{A^2}{\rho r^3} - \frac{\partial Q_r}{\partial r} , \quad (2.62)$$

$$\frac{\partial T}{\partial t} + \nabla \cdot (Tv) = -\rho \frac{\partial \Phi}{\partial z} - \frac{\partial P}{\partial z} - \frac{\partial Q_z}{\partial z} , \quad (2.63)$$

$$\frac{\partial A}{\partial t} + \nabla \cdot (Av) = -\rho \frac{\partial \Phi}{\partial \theta} - \frac{\partial P}{\partial \theta} - \frac{\partial Q_\theta}{\partial \theta} , \quad (2.64)$$

where

$$Q_r \equiv C_2 \rho \delta v_r \text{Min}[\delta v_r, 0] \quad (2.65)$$

and $\delta v_r = v_r(r + \delta r) - v_r(r)$;

$$Q_z \equiv C_2 \rho \delta v_z \text{Min}[\delta v_z, 0] \quad (2.66)$$

and $\delta v_z = v_z(z + \delta z) - v_z(z)$; and

$$Q_\theta \equiv C_2 \rho \delta v_\theta \text{Min}[\delta v_\theta, 0] \quad (2.67)$$

and $\delta v_\theta = v_\theta(\theta + \delta \theta) - v_\theta(\theta)$.
The two alternate equations of energy (2.15) and (2.16) also acquire an additional dissipative source term, \(-Q : \nabla v\), becoming

\[
\frac{d\varepsilon}{dt} + \gamma \varepsilon \nabla \cdot v = \Gamma - \Lambda + \nabla \cdot (C_T \nabla T) - Q : \nabla v \tag{2.68}
\]

and

\[
\frac{\partial \varepsilon^{1/\gamma}}{\partial t} + \nabla \cdot (\varepsilon^{1/\gamma} v) = \frac{1}{\gamma} \varepsilon^{-1} \left[ \Gamma - \Lambda + \nabla \cdot (C_T \nabla T) - Q : \nabla v \right]. \tag{2.69}
\]

Besides appearing in the equation of motion (2.61) and in the energy equations (2.68) or (2.69), \(Q\) will affect the Courant time. Since \(Q\) is only nonzero at shocks, \(Q\) can in principle be used as a diagnostic to map out shock fronts.

2.10 COURANT LIMIT

In any explicit, hydrodynamic code, the time step is selected by the Courant limit. In my second-order, explicit Eulerian hydrocode with artificial viscosity the time step \(\delta t\) is

\[
\delta t < \text{Min} \left[ \frac{\delta x}{v_s(r, z, \theta) + v_z(r, z, \theta) + v_{y_x}(r, z, \theta)} \right] \quad \forall r, z, \theta, \tag{2.70}
\]

where \(\delta x = \delta r, \delta z, \delta \theta, x = r, z, \theta, v_s\) is the velocity of sound, \(v_z\) is the flow velocity in the \(x\) direction, and \(v_{y_x}\) is the artificial viscous velocity in the \(x\) direction. In my code \(\delta t\) is set at one-half the minimum value defined in equation (2.70).
The artificial viscous velocity is:

\[ v_{\nu x} \equiv 4\sqrt{C_2 Q_{xx}/\rho} \]  \hspace{1cm} (2.71)

I derived this from the fact that an explicit diffusion scheme limits the time step by \( \delta t < (\delta x)^2/(4\nu_x) \), where \( \nu_x \) is the kinematic viscosity. Then, by an analogy with the Navier-Stokes equation, \( \nu_z = \delta x (C_2 Q_{zz}/\rho)^{1/2} \).
3. THE COMPUTER CODE ITSELF

3.1 A Flow Chart in Words

In the previous chapter, the basic hydrodynamic equations were discussed. The FORTRAN hydrocode is actually listed in Appendix C, where it is broken down into four sections, each of which is a collection of subroutines. The collection of subroutines are called MAININ, MAINCN, HYD3D2O, and POTS. These, together with the subroutine collection BLKTRI and FFTPACK, which are not listed in Appendix C, and one function called EPMACH which is referenced in BLKTRI, complete the hydrocode. A description of the function of each subroutine collection and each of the subroutines that it contains follows. The program is then described as a whole.

MAININ and MAINCN

The subroutine collection MAININ is used only to input an initially axisymmetric model. It takes an initial axisymmetric model and runs it for the specified number of time steps. It outputs a model file named MOD13 in unformatted binary form, which can be restarted by the subroutine collection MAINCN. It also outputs an extensive formatted diagnostic file named DIAG07. If something is going to go wrong with a model it generally goes wrong in the first 25 time steps, and it is good to look at the DIAG07 file in detail to see if everything is going well before investing more computer time to continue the evolution. The subroutines that are members of MAININ are main$, RITE, SETUP, IZUMI, GEM, and SETUP2. Either MAININ or MAINCN, not both, are used, depending upon whether a given evolution is just being started or is being continued.
The subroutine collection MAINCN is used to continue an already started model. MAINCN takes an already outputted model, from either a MAININ run or an earlier MAINCN run, and runs it for the specified number of time steps. It outputs a model file named MOD13 in unformatted binary form, just as MAININ does. The file called DIAG07 in MAINCN gives only brief model summaries every 50 time steps. The subroutines which are members of MAINCN are main$, RITE, SETUP, GEM, and SETUP2.

main$: The routine main$ is the main program. It contains all the file handling OPEN statements and the main time-step loop. Basic program flow can be understood in detail by looking at main$. The significant difference between the two different versions of main$ in MAININ and MAINCN is in file handling.

RITE: The subroutine RITE is responsible for writing all the output, with the exception of a few debugging statements scattered elsewhere in the program—these are generally commented out. RITE is different in MAININ and MAINCN principally because of the more extensive diagnostics written from MAININ. Both RITE subroutines currently write two more files. A file named MOD12 is a formatted file that contains a snapshot of the model's dynamical variables at the end of the evolution; it is used to do graphics. A file named COEF25 is a formatted file that contains a record of the time-evolving Fourier modes of the density in the equatorial plane. More is said about these Fourier modes in the next chapter. The two files, MOD12 and COEF25, serve as the primary input files to much of the data analysis procedures.
**SETUP:** The subroutine SETUP is responsible for setting up the initial model, and it differs in MAININ and MAINCN essentially by calling IZUMI in the case of MAININ.

**IZUMI:** Subroutine IZUMI in Appendix C reads in a two-dimensional, equilibrium rotating polytrope generated by Izumi Hachisu. I have a version of IZUMI, not shown in Appendix C, that reads in accretion tori with central point masses. When central point masses are used, an additional subroutine called CENTRL is used to add the point mass to the self-gravitating disk potential calculated in POTS; the call to CENTRL must be inserted in main$. I also have a subroutine called DURISE, which reads in centrally condensed polytropes generated by Dick Durisen.

**GEM and SETUP2:** GEM and SETUP2 are the same in both MAININ and MAINCN. They are called only once, to set up some geometric factors that are used to do fluxing and sourcing in HYD3D2O.

**HYD3D2O**

The subroutine collection HYD3D2O is completely written by me. This is where the hydrodynamic equations are solved in second order. It is in HYD3D2O and in POTS, the potential solver, where the greatest part of the computer run time is used. The subroutines that are members of HYD3D2O are DELTA, SOST, SOA, VIS, HYDRO3, EOC, VEL, VLIMIT, STATE, and VRPOW.

**DELTA:** The subroutine DELTA calculates the time step $\delta t$ by using the prescription in equation (2.70) every time step.

**SOST and SOA:** The subroutine SOST sources both of the variables $S$ and
The subroutine SOA sources only $A$. Both of these subroutines use central differences to approximate the partial derivatives that appear in the source terms $S_S$, $S_T$, and $S_A$; as is well known, on a uniformly spaced grid, central differences are second-order accurate in approximating partial derivatives. The only thing the least bit clever in these subroutines is the use of CVMGPI, a statement function that keeps the $A^2/(\rho r^3)$ term in $S_S$ from blowing up if $\rho$ is extremely small. A clever trick of taking an angular derivative by Fourier transforming, multiplying by the mode number, and inverse Fourier transforming was done for a short while in this subroutine, but since it gave the same answers even when there was significant variation azimuthally, it is not presently used.

VIS: The subroutine VIS implements artificial viscosity, as explained in §2.9. VIS first calculates the quadratic diagonal artificial viscosity tensor according to equations (2.65)-(2.67). It then sources $S$, $T$, and $A$ by taking the gradient of this tensor as prescribed by equations (2.62)-(2.64).

HYDRO3 and EOC: The subroutine HYDRO3 fluxes $S$ and $T$ and the subroutine EOC fluxes $A$ and $\rho$. Both use the van Leer second-order monotonic interpolation and make extensive use of conditional vector merge functions (see below). The difference in complexity between equations (2.37)-(2.39) for first-order accurate fluxes and equations (2.42)-(2.47) for second-order accurate fluxes points to the efficiency in using conditional vector merges. A first-order code would be expected to use much less run time than a second-order code dealing with the same problem: however, Tohline's first-order code, which currently does not use conditional vector merges, is only a
little faster than my second-order code, which does. If Tohline's first-order code were rewritten using conditional vector merges it would execute much faster than my second-order code. Only subroutines HYDRO3 and EOC in Tohline's first-order code would have to be rewritten. This is, incidentally, the only place his code is not already second-order when the grid spacing in $z$ and $r$ are uniform.

**VEL:** By using equations (2.34)-(2.36), the subroutine VEL simply calculates the linear velocities that are consistent with the updated values of both the momentum densities and the mass density. A call to VEL must follow both a call to SOST/SOA and a call to HYDRO3/EOC.

**VLIMIT:** The subroutine VLIMIT limits the velocity when $\rho$ is so small as to be vacuum.

**STATE:** The subroutine STATE currently calculates the pressure according to $P = K \rho^\gamma$. It calls a subroutine VRPOW, which takes a vector—the entire array—raised to a real power. VRPOW is FPS specific, but a similar subroutine could be written for any machine. The naive FORTRAN 77 implementation of $P = K \rho^\gamma$ is commented out, but it could easily be restored. VRPOW resulted in quite a bit of run time saving within STATE. If $\gamma$ is not an integer—as it seldom is—most compilers will calculate $\rho^\gamma$ as $e^{\gamma \ln \rho}$. It is useful to force the computer to do this operation at vector speeds.

**POTS**

The subroutine collection POTS, developed by Tohline, solves Poisson's equa-
tion. The subroutines that are members of POTS are SETBDY, SORT, BDYGEN, POT3 and ZAXPHI.

**SETBDY and SORT:** The subroutine SETBDY sets up the arrays used in BDYGEN. SETBDY is called only once in a run of so many time steps. The subroutine SORT is used by SETBDY in order to sort an array that contains the radii of the points of the grid in spherical coordinates.

**BDYGEN:** The subroutine BDYGEN calculates the value of the gravitational potential at the boundary of the grid. BDYGEN is called from main$ every time step to update the value of the potential on the boundary.

**POT3:** The subroutine POT3 is the main calculation engine of the potential solver. POT3 is called every time step from main$.

**FFTPACK and BLKTRI:** The subroutine collections FFTPACK and BLKTRI are both called within POT3. FFTPACK is used to Fourier transform $\Phi$ on the boundary and $\rho$ everywhere into $L_{max}$ Fourier components $\Phi_m$ on the boundary and $\rho_m$ everywhere. The dynamical variables are discretized to have $L_{max}$ azimuthal zones. BLKTRI is then used to solve the linear equations that result from discretizing the 2-D Helmholtz equation

$$
\frac{\partial^2 \Phi_m(r, z)}{\partial r^2} + \frac{1}{r} \frac{\partial \Phi_m(r, z)}{\partial r} + \frac{\partial^2 \Phi_m(r, z)}{\partial z^2} = 4\pi G \rho_m(r, z). \quad (3.1)
$$

The boundary $\Phi_m$ and source $\rho_m$ everywhere are fed into BLKTRI, which generates $\Phi_m$ everywhere. FFTPACK is again called, this time to inverse Fourier transform $\Phi_m(r, z)$ everywhere to $\Phi(r, z, \theta)$ everywhere. This is how new $\Phi$'s are generated from new $\rho$'s. The references on this are Swartztrauber and Sweet (1975) and Swartztrauber (1974).
**ZAXPHI:** The subroutine ZAXPHI calculates values of the potential on the $z$ axis by a suitable interpolation. Values of the potential on the $z$ axis are needed in Tohline's code since he centers $S$ and $T$ on the vertices of the $\tau_p$ control volume. Since I center $S$, $T$, $A$, and $\rho$ in the same place, ZAXPHI need not be called.

Now that all the subroutines have been named and their functions described, the following is what is actually done. The program is run for a specified number of time steps. The model is read in and the grid and all the variables initialized. Then, in the time step loop in subroutine main$, the subroutines are called in the order DELTA, SOST, SOA, VIS, VEL, HYDRO3, EOC, VEL, STATE, BHYGEN, POT3, and then RITE. Every 50 time steps, another RITE is called which outputs a model summary. After the last time step, several calls to RITE are made, and this writes out the binary model file for restarting the model and the formatted model file for graphic interpretation.

3.2 **COMPUTERS TODAY**

Numerical hydrodynamics is run on real computers that exist today with existing compilers. At the moment, and probably for some time in the future, that means FORTRAN. Computer scientists and others are fond of poking fun at FORTRAN, but the fact that the world has billions of dollars invested in FORTRAN programs assures that the language will be with us for some time to come. FORTRAN does have defects, but it is a dynamic language, as can be gathered from a quote from Seymour Cray. When he was asked a few years ago about what language his future machines would be programmed in, he is
rumored to have replied, "I do not know what language it will be, but it will be called FORTRAN." The new FORTRAN SX standard may even be out next year. Explicit vector syntax and the possible abolition of COMMON blocks are some of the things promised. How much, if any, large-scale parallelism in the form of parallel processing will be incorporated into the new standard is anyone's guess.

I was lucky enough to spend August of 1986 attending the University of Minnesota Supercomputer Institute Summer Workshop sponsored by NSF. Included in these four weeks, besides constant access to a Cray-2, was a course given by Pacific-Sierra Research Corporation. Their text, Efficient FORTRAN Techniques for Vector Processors (Pacific-Sierra Research 1984), has guided my subsequent FORTRAN code writing. The rest of this section is written in stream of consciousness and should be viewed as an experimental, ephemeral, literary work.

The most important rule I have learned is to avoid I/O when ever possible. If the problem becomes I/O bound and not computationally bound, then the computation will move at the speed of sound (the physically limiting factor on a disk drive) instead of the speed of light (the physically limiting factor within the CPU). When forced to do I/O, as much of it as possible should be done in unformatted form. Writing arrays out by just listing the array variable with no subscripts saves computer time in formatted or unformatted I/O. (This last trick runs into problems on IBM's MVS operating system because the length of the largest available logical record length is limited.)

The next most important thing that I have learned is to be very careful about what goes into triply nested DO loops. In a 3-D hydrocode, one obvi-
ously has to do things in all three dimensions. Within nested DO loops, the longest dimension should be the leftmost subscript (rightmost if writing Pascal). If possible, divisions should not be done within frequently executed DO loops. Division currently takes more cycles than multiplication and addition. On Crays, divisions may not be done to 64 bits precision unless you specify that precision as a compiler option; current default is 46 bits (Cray 1986). Of course, this makes division even more expensive. The FORTRAN manual produced by the manufacturer discloses computer peculiarities. For instance, the Cray-2 has a hardware square root, but a software divide. Many manufacturers have special libraries of functions that vectorize on entire arrays. Specially tailored FORTRAN that is machine dependent can be put in statement functions that are written as in-line code when the program is compiled. If there are several different versions of the same statement function designed for different machines, the code will still be transportable. (You can never tell where and on what computer system you will have to run your program next.) To change machines, merely remove C's in column one. The INCLUDE statement in many operating systems also offers another solution to portability, but not every system will have this feature. Subroutines or functions (except statement functions) should not be called from within frequently used, nested DO loops. DO loops should be short; if the programmer cannot read them, imagine the problem the compiler must have. No current FORTRAN compiler—despite the claim of the manufacturers—is artificially intelligent. IF statements should be avoided within nested DO loops. IBM, some of the Japanese manufactures, and Cray, too, provide compilers that are currently smart enough to replace extremely simple and obvious IF statements.
with conditional store statements. Be forewarned that all the manufacturers use
different catch phrases for the same thing. If the compiler has a flow trace op-
tion to time subroutines or an execution analyzer, programming effort can be
concentrated on the areas where execution time is being spent.

3.3 Conditional Vector Merge and Bit Mask Registers

The heart of second-order van Leer monotonic interpolation is contained in
equations (2.40) and (2.41). To naively implement this in FORTRAN would
require nested IF statements. Since this must be done in all three dimensions
everywhere, triply nested DO loops are required. Equations (2.42)-(2.47) are
what actually has to be implemented in a discrete cylindrical grid. Because of
their generally unvectorizable nature, IF statements would be murder at exe-
cution time; the computer budget would needlessly go through the roof. The
way around this is to use conditional vector merge functions. Conditional vector
merge functions CVMG(something) are Cray extensions to FORTRAN (Cray
capabilities exist in CDC’s machines, and in its subsidiary ETA, called bit mask-
ing (CDC 1981), and conditional vector merge functions can be made to exit on
other machines like the FPS’s and others (Williams and Tohline 1987 c). Even
on a scalar machine like a VAX 11/750, conditional vector merges can be created
in a statement function. Of course, on a scalar machine nothing is gained, but
nothing is lost either, and portability is maintained.

All conditional vector merge functions take three arguments. Depending
upon the truth or numerical value of the third argument, either the first or second
argument is stored. Computation can be done within the first two arguments. In fact, it can be done within the third argument, also. Conditional vector merge functions are conditional store functions, and the reason they are more useful than a conditional branch statement (like an IF statement) is tied up with the logic of local memory cache, pipelines, and vector registers. The physics behind this is that there is less travel time when there is less distance. We physicists should exploit this engineering to solve our physics problems more efficiently.

The conditional vector merge functions CVMGP and CVMGZ can be understood in terms of IF statements as:

\[
J = \text{CVMGP}(A,B,C) := \begin{cases} 
J = B & \text{IF } (C \geq 0.0) \\
J = A & \text{IF } (C < 0.0) 
\end{cases}
\]

\[
J = \text{CVMGZ}(A,B,C) := \begin{cases} 
J = B & \text{IF } (C \approx 0.0) \\
J = A & \text{IF } (C \approx 0.0) 
\end{cases}
\]

Of course, it is syntactically incorrect in FORTRAN to have a statement function on two lines, except as a continuation, and the whole purpose of using CVMG(something) is to avoid IF statements anyway. Nevertheless, the above examples clearly demonstrate what these CVMG's are. The following are code fragments taken from Appendix C that illustrate the use of conditional vector merge functions.

```
C* **** * * * * * * * * * *Van Leer Slopes for RHO* **** * * * * * * * * *
SLOPER(F,B,C) = CVMGP(1.0,0.,(B-C)*(F-B))*2.*(B-C)*(F-B) /
   1  CVMGZ(1.,F-C,F-C)
```
Van Leer Interpolation for RHO, A, S, T, EPS*(1/GAMMA)***

\[ VLI(Q, Q_1, DX, V, SLOPE, SLOPE_1) = \]
\[ 1 \; CVMGP(Q_1 + (DX - V \times HALFDT) \times SLOPE_1 / (2. \times DX), \]
\[ 2 \; Q + (DX + V \times HALFDT) \times SLOPE_1 / (2. \times DX), V) \]

Inside a triply nested loop in L, K, J:

\[ J_P = J + 1 \]
\[ J_M = J - 1 \]
\[ UTIL(J, K, L) = SLOPER(RHO(J_P, K, L), RHO(J, K, L), RHO(J_M, K, L)) \]

Inside a triply nested loop in L, K, J:

\[ J_M = J - 1 \]
\[ TEMP = U(J, K, L) \]
\[ FR(J, K, L) = VLI(RHO(J, K, L), RHO(J_M, K, L), DR, TEMP, \]
\[ 1 \; UTIL(J, K, L), UTIL(J_M, K, L)) \times TEMP \times GEMR \times (R(J) - TEMP \times HALFDT) \]

For an implementation of these CVMGs on different machines, the reader is referred to subroutine EOC or HYDRO3 in section HYD3D2O of Appendix C.
4. ANALYZING DATA

4.1 Graph Everything

During an evolution millions of numbers are generated each time step. However, the purpose of doing the simulation is to understand what is happening, not to generate millions of numbers. To make sense out of the dynamical variables, which are 3-D arrays evolved in time, is a formidable task in itself. Obviously, looking at printed output of the variables is a hopeless task. Traditional 2-D hydrodynamics has relied on contour plots and vector stream plots. Winkler and Norman (1986) describe a color graphics system called Munacolor. Working implementations of this system evidently now exist at the Los Alamos National Laboratory, the University of Minnesota Supercomputer Institute, and the National Center for Supercomputer Applications at the University of Illinois at Urbana-Champaign (Winkler et al. 1987). These systems have extraordinarily high I/O bandwidths, but at least an order of magnitude higher is needed before the human visual system is saturated. The article by Winkler et al. mentions the visualization problems and predicts that the I/O bandwidth needs to be two orders of magnitude higher for 3-D hydrodynamics before the human visual system is saturated. With systems like this, one imagines a Cray X-MP as a front end for a massively parallel machine like the Connection Machine from Thinking Machines Corporation.

At present, though, for those of us that are working with realized machines, analyzing the data is not so automated. In 3-D hydrodynamics, contour plots and vector stream plots are still useful, but a plane must be chosen in three-
dimensional space. Another useful way of presenting the data is by using iso-
density surfaces. Given the values of a dynamic variable like mass density $\rho$ on
a discrete grid in three-dimensional space, an isodensity surface is the surface in
3-D that encloses all the volume where the density is above a certain value. Nat-
urally, a well-designed graphics system creates a proper smooth surface, making
the obvious interpolations. At present we use two different systems that are ca-
pable of drawing isodensity surfaces. Every graphics package in the world seems
to handle contour plots and vector stream plots, but packages that draw isoden-
sity surfaces are still somewhat rare. The NCAR plotting package is one such
system, and it currently runs on the campus mainframe IBM 3084QX4 running
MVS/XA/tso and on both VAX 11/750s running VMS in the physics and astron-
omy department. In Appendices A and B, many isodensity surfaces are shown.
In fact, not a single contour plot appears in either of these papers, even though
we have examined contour plots of $\rho$ with momentum density vectors superim-
posed on the contour lines along every conceivable direction. For example, we
have examined contour plots with $\rho(r,z = \text{constant}, \theta)$ and $\rho(r,z, \theta = \text{constant})$
with every conceivable value of $z$ and $\theta$ held constant. At least for black-on-white
images in a scientific journal, families of 3-D isodensity surfaces seem to convey
information without distraction and are easier to discuss than the plethora of
contour plots that would be needed to do the same task. By showing a series of
isodensity surfaces at different values of the density but at the same time, the
density structure of a model at an instant can be ascertained. By showing a
series of isodensity surfaces in time, the time evolution of a model can be seen.

The other isodensity surface system enables us to have contour plots on the
same picture with the isodensity surfaces, and it uses an image processor that is attached to the astronomy VAX 11/750. This system is discussed in §4.3.

4.2 Fourier Analysis and Spectral Methods

An extremely useful way of analyzing the growth rate and pattern speed of nonaxisymmetric structures is to Fourier transform the density in angle space. In our particular application, the Fourier transforms are done at no additional computer cost because, at each time step during an evolution, the variable \( \rho \) is already Fourier transformed in the azimuthal direction as standard input in to the Poisson solver (see §3.1).

On our discrete grid, henceforth the location of a grid zone in the coordinate \((r, z, \theta)\) will be referred to by its respective integer values \((J, K, L)\). The grid boundaries extend to \((J_{\text{max}}, K_{\text{max}}, L_{\text{max}})\). If there is an even number of angular zones, \(L_{\text{max}}\), and no angular symmetry, then the discrete Fourier cosine coefficients are

\[
a_m(J, K) \equiv \frac{1}{\pi} \sum_{L=1}^{L_{\text{max}}/2} \rho(J, L, K) \cos(mL\delta\theta)\delta\theta ,
\]

and the Fourier sine coefficients are

\[
b_m(J, K) \equiv \frac{1}{\pi} \sum_{L=1}^{L_{\text{max}}/2} \rho(J, L, K) \sin(mL\delta\theta)\delta\theta ,
\]

where \(m = 0, 1, 2, \cdots, L_{\text{max}}\) and \(\delta\theta \equiv \frac{2\pi}{L_{\text{max}}}\).

All of the simulations we have published so far use "\(\pi\) symmetry." If only even azimuthal modes are expected to develop, this will cut computer storage in
half and computer execution in half. The phrase "\( \pi \) symmetry" in the angular coordinate means that the dynamical variables have a periodic structure such that they are the same in the range \( 0 \leq \theta < \pi \) as they are over the range \( \pi \leq \theta < 2\pi \).

If there is an even number of unique angular zones and "\( \pi \) symmetry" is enforced, then the discrete Fourier cosine coefficients are

\[
a_m(J, K) \equiv \frac{2}{\pi} \sum_{L=1}^{L_{\text{max}}} \rho(J, L, K) \cos(mL\delta\theta)\delta\theta ,
\]

and the Fourier sine coefficients are

\[
b_m(J, K) \equiv \frac{2}{\pi} \sum_{L=1}^{L_{\text{max}}} \rho(J, L, K) \sin(mL\delta\theta)\delta\theta ,
\]

where \( m = 0, 2, 4, \ldots, L_{\text{max}} \) and \( \delta\theta \equiv \frac{\pi}{L_{\text{max}}} \).

For convenience, one can define

\[
c_m(J, K) \equiv \sqrt{a_m(J, K)^2 + b_m(J, K)^2},
\]

and

\[
\phi_m(J, K) \equiv \arctan \frac{b_m(J, K)}{a_m(J, K)},
\]

so that the density may be reconstructed as

\[
\rho(J, K, L) = \frac{c_0(J, K)}{2} + \sum_{m=1 \text{ or } 2}^{L_{\text{max}}} c_m(J, K) \cos[mL\delta\theta + \phi_m(J, K)],
\]

where \( (m = 1 \text{ or } 2) \) means summing over all integers for no angular symmetry and summing over even integers for "\( \pi \) symmetry". It is useful to think of the
density as being resolved into $\rho_m$ structures by the equation

$$\rho(J, K, L) = \frac{c_0(J, K)}{2} + \sum_{m=1 \text{ or } 2}^{L_{\text{max}}} \rho_m(J, K) ,$$  

(4.8)

where the $\rho_m$ structure is given by

$$\rho_m(J, K) \equiv c_m(J, K) \cos[mL\delta\theta + \phi_m(J, K)] .$$  

(4.9)

For $m > 0$, $c_m$ is the amount of the $m^{th}$ mode so the exponential growth rate of the $m^{th}$ mode is $G_m = \partial \ln c_m / \partial t$. The angle at which the $m^{th}$ mode points is $\phi_m$. The pattern speed at which the $m^{th}$ mode points is

$$W_m \equiv -\frac{1}{m} \frac{\partial \phi_m}{\partial t} .$$  

(4.10)

The above equation can be easily demonstrated by calculating the rate, i.e., the time derivative, for the maximum of equation (4.9); see Appendix A.

A linear tensor virial equation analysis (Chandrasekhar 1969; Tassoul and Ostriker 1970; Ostriker and Bodenheimer 1973; and Durisen 1975) predicts values for the growth rate of the bar-mode $G_2$ and the pattern speed $W_2$ for distortions from axisymmetric equilibrium of the form

$$\delta \rho \propto -r \frac{\partial \rho_A}{\partial r} e^{i2(\theta + W_2 t)e^{-iG_2}} ,$$  

(4.11)

where $\rho_A$ is the axisymmetric equilibrium density. These numbers, $W_2$ and $G_2$, can be compared to the prediction of the linear tensor virial equation. $W_2$ is just one-half the slope of the $\phi_2$ versus time graph and $G_2$ is just the slope of the
In $c_2$ versus time graph. To show that these are global functions across the entire model, these quantities are graphed at every radii $r$. To illustrate the constancy of $G_2$ and $W_2$, surfaces of $\phi_2$ versus $r$ and time can be plotted and surfaces of $\ln D_2$ versus $r$ and time can be plotted as is done in Appendix A. The normalized amplitude $D_m$ is defined as

$$D_m = \frac{c_m}{\sum_{i=0}^{L_{\text{max}}} c_i}. \quad (4.12)$$

In the linear regime $\sum_{i=0}^{L_{\text{max}}} c_i \approx c_0$ and, since $c_0$ is essentially constant in time while the nonaxisymmetric distortion is small, then $\partial \ln D_m/\partial t \approx \partial \ln c_m/\partial t$. The linear tensor virial equation analysis is only valid for small distortions from axisymmetric equilibrium, anyway.

The normalized amplitude is a better variable than $c_m$ when the model becomes highly nonaxisymmetric, since $D_m$ is a better measure of how much of the density is in the $m^{\text{th}}$ mode. Notice that $D_m$ and $\phi_m$ are, themselves, functions of two variables $r$ and $z$. In this work, to date, $D_m$ and $\phi_m$ have been graphed only on the equatorial plane, $z = 0$.

4.3 Image Everything

To better analyze a 3-D model, it is best to image as much of it as possible so we can understand what is going on. The 3-D simulations are very computer intensive, and so far the computer cost of the graphics has been minimal. It takes a few CPU days on the FPS M64/60 to run a model. It takes only a few hours of CPU time on the IBM 3084QX4 to graph everything in nearly every conceivable
way. One of the problems with outputting data to the Benson plotter, though, is that it is very noninteractive. The worst problem is that you cannot change the contour levels or the value of the density that fixes a particular isodensity surface. With an image processor all of this changes. Tohline realized that the Astronomical Image Processing System (AIPS), designed for radio astronomy, could be used to produce contour images of our models. The first images that we produced using AIPS were contour plots of the density in the equatorial plane. We have made a video movie of the early evolution, from axisymmetry to when nonaxisymmetric spirals become apparent. At the end of this movie some isodensity surfaces of the end of the evolution are shown; this portion was produced by using a modified version of Display82, a system of programs for displaying 3-D information in computerized tomography (CT) data (Udapa 1983 and Frieder 1983). The Display82 system was provided for us by Gabor Herman of the Department of Radiology, Hospital of the University of Pennsylvania.

The system we are currently using, called STAR4, which was developed by Monika Lee from Display82 and bits and pieces from elsewhere, allows four images to be displayed at once. On Figure 4.1 is a black-and-white image of what appears on the image processor screen as a color image. Three of the images are color contour maps. The upper left is $\rho(r, z = 0, \theta)$, the upper right is $\rho(r, z, \theta = 0)$, the lower left is $\rho(r, z, \theta = \pi/2)$, and the lower right is two different isodensity surfaces. The inner isodensity surface is the high-density surface and the outer isodensity surface is the low-density surface. The number on the figure is the time in terms of central rotations. This model is the end of the evolution for the $n = 1.8$ polytrope described in Appendix B. The triaxial central object is visible
as the high density region within the circumstellar ring.
Figure 4.1.—The $n = 1.8$ polytrope with $q = 3/2$ and $\beta_i = 0.310$ at a time of 19.42 central rotation periods. The upper left is $\rho(r, z = 0, \theta)$, the upper right is $\rho(r, z, \theta = 0)$, the lower left is $\rho(r, z, \theta = \pi/2)$, and the lower right is two different isodensity surfaces. The inner isodensity surface is the high-density surface and the outer isodensity surface is the low-density surface.
5. MODELS RUN IN THE SECOND-ORDER CODE: 
DESCRIPTION AND PRELIMINARY CONCLUSIONS

5.1 General Description

The principle reason a second-order hydrodynamic code was written even though a well-tested, first-order hydrodynamic code was available is because of the numerical diffusion in any first-order scheme. When studying nonaxisymmetric dynamic instabilities, this numerical diffusion manifests itself by being Fourier mode dependent; see equation (14) in Appendix A. Numerical diffusion damps the growth rate of higher-$m$ $\rho_m$ structures; see equation (4.9). To investigate what happens when $\beta_i$ is nearer $\beta_d$, a nondiffusive code is required since $G_2$, the bar-mode growth rate, is damped.

I have run several initially axisymmetric models in the second-order code. Only two of these models have been run in both Tohline’s first-order code and my second-order code. The nonrotating model, or virtually nonrotating model, was run in both codes. The nonrotating model’s importance in convincing us that sourcing and fluxing need to be done separately has been described in §2.4. Results from the nonrotating model will be described in the next section, §5.1.

The other simulation that was run in both codes was the $n = 1.8$, $\beta_i = 0.310$, and $q = 3/2$ model. Both of these simulations were done before either Tohline’s first-order code or my second-order code were altered to perform sourcing and fluxing as separate operations. Figure 5.1 shows results from this simulation as run in Tohline’s first-order code at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$, all
versus time in polytropic units. The solid line segment on Figure 5.1 is a line with a slope equal to the growth rate predicted by a linear tensor virial equation analysis; this line is also coincident with the slope of the $D_2$ line during its linear growth phase, corrected by numerical viscosity as detailed in equation (14) of Appendix A. Figure 5.2 shows this simulation run in my second-order code with everything similarly defined. The slope of the solid line segments are again taken from predicted growth rates derived from a linear tensor virial equation analysis. The second-order code *performs as expected*, exhibiting a growth rates very close to the predictions of the linear theory, but the $D_2$ curve is so noisy that the coincidence is not immediately apparent. When sourcing is done separately from fluxing, a cleaner growth rate appears; see §5.3. In the nondiffusive second-order code, no correction for growth rates is permitted or needed!

In the second-order hydrocode, a delicate and complicated interplay of power going from one Fourier mode to another is revealed. In Figure 5.2 during the linear growth regime, notice that when $D_2$ decreases, $D_4$ increases as well. Contrast this to the decrease (damping) of the higher mode amplitudes $D_4$ and $D_6$ in Figure 5.1 (the first-order code) during the time period $t = 1000-2000$ in polytropic units. (Polytropic units are explained in Appendix A.)

Except for one brief interval of time, centered at $t = 1500$, during which $D_4 > D_2$ in the second-order simulation, the isodensity surfaces in the first-order and second-order simulations are indistinguishable. When $D_4 > D_2$, in the second-order simulation the isodensity surfaces are somewhat boxlike, rather than nearly axisymmetric as they are in the first-order simulations when $D_2$ is this small.
Figure 5.1.—The $n = 1.8$, $\beta_i = 0.310$, and $q = 3/2$ model simulation run in Tohline’s first-order code at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$, all versus time in polytropic units. The solid line segment is a line with a slope equal to the growth rate predicted by the linear tensor virial equation analysis.
Figure 5.2.—The $n = 1.8$, $\beta_i = 0.310$, and $q = 3/2$ model simulation run in my second-order code at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$, all versus time in polytropic units. The two solid line segments are lines with a slope equal to the rate predicted by the linear tensor virial equation analysis.
It is interesting to note that, after the model has made its brief excursion with the $D_4$ distortion, the $D_2$ distortion starts to grow again with the same growth rate it had before $D_4$ increased: however, given the noisy nature of this simulation perhaps the perception of sameness is the result of wishful thinking. A similar phenomenon is apparent in the $n = 1.8$, $\beta_i = 0.313$, $q = 3/2$ model evolution shown in §5.3.

Several models were run when the second-order hydrocode was being developed and have given me confidence in the general soundness of the code. An $n = 1.5$, $\beta_i = 0.382$, $q = 3/2$ model with and without random initial perturbations was run. In an exponentially growing instability, these runs convinced me that a random perturbation is unnecessary. The simulations run in Appendix A, for instance, were randomly perturbed at the level of $5 \times 10^{-4}$. Either the machine noise perturbs the model, or the initial equilibrium itself is slightly perturbed because of interpolation error. (Interpolation error is caused by the fact that the equilibria are calculated in another grid and interpolated into the cylindrical grid of the hydrocode. There is interpolation error, for example, which is second order and proportional to the second derivative of the density and specific angular momentum with respect to the spatial distances.)

The $\beta_i = 0.382$ model was also run for an extended evolution with the potential solver called only once at the very outset of the evolution. This helped convince us that the potential solver itself was not causing any numerical instabilities. A high $\beta_i$ model was in this test evolution because it was extremely dynamically unstable and, therefore, nonaxisymmetric structures were expected to develop very quickly. A relatively exhaustive test of all features of the 3-
D hydrodynamic code could thereby be accomplished using a reasonably small amount of computer time.

Since we started sourcing and fluxing separately, only two models have been run in the second-order code. These are the \(n = 1.8, \beta_i = 0.313, q = 3/2\) model, briefly reported in §5.3, and the \(n = 0.8, \beta_i = 0.313, q = 3/2\) model not yet reported. I literally had to stop computing on these models so I could write my dissertation.

5.2 Model with \(n = 1.5, \beta_i = 8.9 \times 10^{-5}, q = 3/2\)

The virtually nonrotating model, \(n = 1.5, \beta_i = 8.9 \times 10^{-5}\), and \(q = 3/2\), is the model that taught us to source and to flux separately (see §2.4). In this initially spherically symmetric model, pressure balances gravity and nothing should develop dynamically. When we ran the model in Tohline’s first-order code and in my newly developed second-order code, catastrophe developed after time \(6\); see Figure 5.3. This rapid growth of \(D_2, D_4,\) and \(D_6\) after time \(6\) means that this initially axisymmetric model had spontaneously developed nonaxisymmetric structures. There is no reasonable excuse for this. This is purely numerical.

Tohline then sourced \(A\) separately from fluxing. I did the same. The nonaxisymmetric structures disappeared, as is apparent from Figure 5.4. The contour plot of density in the equatorial plane Figure 5.5 shows only a little numerical noise near the center of the grid. However, the \(r-z\) contour plot, Figure 5.6, shows a numerically induced mess in the \(r-z\) plane.

At this point the solution seemed obvious: not just the variable \(A\), but \textit{everything} should be sourced and fluxed separately. As Figure 5.7 shows, after
Figure 5.3.—A result from the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when sourcing and fluxing were done together). $\ln D$ versus time is shown at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$. The instability here is purely numerical.
Figure 5.4.—Results from the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $A$ was sourced and then separately fluxed, but $S$ and $T$ were fluxed and sourced together). $\ln D$ versus time is plotted at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 5.5.—A 2-D contour plot of $\rho(r, z = 0, \theta)$ for the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $A$ was sourced and then separately fluxed, but $S$ and $T$ were fluxed and sourced together) at $t = 20.74$, polytropic time.
Figure 5.6.—A contour plot of $\rho(r, z, \theta = 0)$ for the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $A$ was sourced and then separately fluxed, but $S$ and $T$ were fluxed and sourced together) at $t = 20.74$, polytropic time.
implementing separate sourcing and fluxing in all quantities, again no nonaxisymmetric structures developed. But, more important, structure in the meridional planes never developed! Figures 5.8 and 5.9 are a pleasure to behold: they show, at \( t = 21.0 \) (polytropic units), the density contours exactly like the initial equilibrium figure. Dull has stayed dull! This demonstration indicates that it is not all that easy to design hydrodynamic codes that will do "nothing." Unless they are capable of doing nothing when they have a stable equilibrium model, though, it is dangerous to trust their dynamical evolution of a model where something is expected to happen.

5.3 Model with \( n = 1.8, \beta_i = 0.313, q = 3/2 \)

An \( n = 1.8, \beta_i = 0.313, q = 3/2 \) model has been run in the second-order code since we learned to source and flux separately. The results of the evolution are quite similar to the \( n = 1.8, \beta_i = 0.310, q = 3/2 \) model where sourcing and fluxing were done together, except that the \( D_2 \) amplitudes are less noisy. Figure 5.10 shows the standard plot of \( \ln D \) versus time for this model, again at \( r/R_{eq} = 0.640 \) on the equatorial plane. Notice how, during the linear (small distortion from axisymmetry) regime, the growth rate matches the predictions of the solid line segment, which is the prediction obtained from a linear tensor virial equation analysis. Actually, the line segment here is for a \( \beta_i = 0.310 \), not a \( \beta_i = 0.313 \) like the initial model. The value of the slope of the line segment on Figure 5.10 at \( \beta_i = 0.313 \) actually should be a little steeper and would match the simulation even more closely. The delicate interplay of power going from one Fourier mode to another is also revealed in this model. In fact, when the
Figure 5.7.—Results from the $n = 1.5, \beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $S$, $T$, and $A$ were sourced and then separately $S$, $T$, $A$, and $\rho$ were fluxed). $\ln D$ versus time is shown at $r/R_{eq} = 0.64$ on the equatorial plane. The solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$. 
Figure 5.8.—A 2-D contour plot of $\rho(r, z = 0, \theta)$ for the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $S$, $T$, and $A$ were sourced and then separately $S$, $T$, $A$, and $\rho$ were fluxed) at $t = 21.03$, polytropic time.
Figure 5.9.—A contour plot of $\rho(r, z, \theta = 0)$ for the $n = 1.5$, $\beta_i = 8.9 \times 10^{-5}$, and $q = 3/2$ model simulation run in my second-order code (when $S$, $T$, and $A$ were sourced and then separately $S$, $T$, $A$, and $\rho$ were fluxed) at $t = 21.03$, polytropic time.
amplitudes became this large the same sort of interplay is seen in the first-order runs. Around $t = 6 - 7 t_c$, the $D_4$ mode briefly overtakes the $D_2$ mode. Again, as in the $\beta_i = 0.310$ model run in the second-order code, the isodensity surfaces are boxlike when $D_4 > D_2$. Otherwise, though, the isodensity surfaces are mostly indistinguishable from the first-order simulation of $\beta_i = 0.310$, at least through $14 t_c$.

Clearly the $D_2$ versus time plots are noisier in the second-order results than in the first-order results. However, with sourcing and fluxing done separately, the plots become less noisy than when sourcing and fluxing are done as one operation. How much of the interplay of power between the Fourier modes is physical and how much numerical is yet to be determined. To investigate this problem, some simulations favoring a particular distortion, as opposed to random perturbations or no perturbations, need to be run. There are some interesting nonlinear mathematical problems in physics occurring here. Of course, as far as astronomy is concerned, this problem is probably unimportant—the interstellar medium is probably not perturbed by the eigenmode distortion that would produce a flat growth rate, and the details of higher Fourier mode probably do not affect the overall behavior of the gas cloud.
Figure 5.10.—The $n = 1.8$, $\beta_i = 0.313$, and $q = 3/2$ model simulation run in my second-order code at $r/R_{eq} = 0.64$ on the equatorial plane, where the solid line is $D_2$, the dashed line is $D_4$, and the chained-dashed line is $D_6$ and is versus time in central rotation periods. The solid line segment is a line with a slope equal to the growth rate predicted by a linear tensor virial equation analysis for a $\beta_i = 0.310$ model instead of $\beta_i = 0.313$. The growth rate predicted using the linear tensor virial equation for a $\beta_i = 0.313$ should be a little larger than for a $\beta_i = 0.310$. On this plot, the time units are not given in polytropic units, but in terms of central rotation periods, where $t_c = 2\pi/\omega_c$ and $\omega_c$ is the angular velocity in the center of the model. We now have switched to presenting graphs in terms of central rotation periods because it is more physically significant than polytropic time. (As a reminder, polytropic units are defined by $K = 1$, $G = 1$, and $M = 1$.) Polytropic time units vary wildly from model to model, especially as the degree of compressibility of the gas is changed. Since the gas is rotationally supported, except along the axis of rotation, in a rapidly rotating model $t_c$ obviously provides a natural time scale.
6. SUMMARY

6.1 GRAVITATIONAL-ROTATIONAL DYNAMIC INSTABILITY

The gravitational-rotational dynamic instability that occurs when the ratio of the rotational to gravitational potential energy $\beta > 0.27$ was investigated with a first-order, 3-D, hydrodynamic computer code for models having polytropic indices $n = 0.8, 1.0, 1.3, 1.5, \text{ and } 1.8$. Initially, all models were in axisymmetric equilibrium, they had $\beta_i = 0.310$, and they had a specific angular momentum distribution identical to that of a Maclaurin spheroid (Williams and Tohline 1985, 1987 a). This study was the first to examine models having a range of compressibility at a fixed $\beta_i$; it also analyzed the behavior of models having the lowest $\beta_i$ above $\beta_d$ studied so far. In this study it was found that the growth rate and pattern speed of the bar-mode dynamic instability, as predicted by the linear tensor virial equation, were correct in the linear (small distortion from axisymmetry) regime. In the early nonlinear regime the growth rate and pattern speed remained unchanged. The bar-mode was found to exhibit a two-armed spiral pattern instead of a simple bar. The tightness of the spiral increased with increased polytropic index, i.e., with increased compressibility. The agreement between the predictions obtained from the linear tensor virial equation, which eliminates terms of second or higher degree in the nonaxisymmetric distortion, and the results obtained from the fully nonlinear 3-D hydrodynamic simulation lends credibility to both methods. This is particularly important since direct observations of gas clouds in the quasi-equilibrium phase of star formation are presently impossible to obtain. Higher-order modes, $4\theta$ and $6\theta$, were seen to
grow in the early nonlinear regime, but they were shown to be harmonics of the bar-mode and a necessary ingredient for the $2\theta$ bar-mode to become tighter in angle space. In the initial investigation, calculations ended when the two-armed spiral hit the edge of the calculational grid.

Two of the polytropes, $n = 1.8$ and $n = 0.8$, were further evolved with the radial grid doubled in number and extent, and the calculation was restarted from a time just before the spiral arms hit the old, smaller calculational grid. The same well-tested first-order code written by Tohline was used to continue the evolutions (Williams and Tohline 1987 b, 1987 d). These two models were chosen because they represented the extremes of the compressibility parameter investigated so far. Also, they both represented fair approximations to interesting astrophysical systems, with the $n = 1.8$ polytrope being relevant to star formation and the $n = 0.8$ polytrope relevant to neutron stars. Earlier studies had determined that this dynamical instability leads to a type of fission; specifically, it was found to lead to the formation of a central triaxial, bar surrounded by a ring or disk of material. The ring was found to be separated from the central object by a relatively clear gap. It was suggested that the gap occurred at the corotation radius, that is, where the pattern speed of the central triaxial object equaled the angular velocity of the fluid at that radius. In the calculations reported here (Williams and Tohline 1987 d) the pattern speed of the bar-mode was shown to increase in the spiral arm ejection phase, and thereafter to go through periodic pulsations. In the $n = 1.8$ polytrope, as the two-armed spiral wrapped up on itself and moved back in radially, a shock was found to be apparent in the isodensity surfaces. Finally, the central triaxial object was shown to pulsate and, on a
secular time scale, was observed to continue transporting some of its angular momentum outward to the circumstellar ring material.

6.2 second-Order Hydrodynamics

A second-order, 3-D hydrocode was written by me, patterned after Norman and Winkler's (1986) 2-D hydrocode; a brief description of the new code was given by Williams and Tohline (1987 c). This code was designed to vectorize, to use pipelines, or to do both, depending on the capabilities of the computational machine being used. It was designed to use the same potential solver as Tohline's first-order code because this particular routine was already second-order accurate in the spatial differences. The centering of the dynamical variables was specified differently in the two codes, with the second-order code centering all quantities at cell centers; Tohline's first-order code currently centers $A$ and $\rho$ at cell centers, but places $S$ and $T$ on the vertices of the $A$, $\rho$ cells. The second-order code was tested rigorously. The results of a long evolution on the $n = 1.8$, $\beta_i = 0.313$, $q = 3/2$ model indicated that the growth rate agrees with the predictions of a linear tensor virial equation analysis. The corrections for numerical viscosity that were required by the first-order code were not needed in the second-order code in order to obtain agreement with the linear tensor virial equations. Qualitatively, the isodensity surfaces were found to be nearly indistinguishable between the first- and second-order codes. In the evolution that was modeled using the second-order code, higher order Fourier modes were observed not to damp in the early linear evolution, as they had done in the analogous first-order code evolution.

The importance of separating the operations of sourcing and fluxing was
discovered while following the evolution of a very slowly rotating model. As a result of this discovery, the structure of both the first- and second-order codes was modified so that, in all future evolutions, these operations will be split.
7. FUTURE DIRECTIONS

7.1 DIFFERENT ROTATION LAWS

All centrally condensed, rapidly rotating ($\beta_i > \beta_d = 0.27$) initially axisymmetric equilibria that have been studied (Durisen and Tohline 1980, 1985; Tohline, Durisen and McCollough 1985; Durisen and Gingold 1986; Durisen, Gingold, Tohline, and Boss 1986; Williams and Tohline 1987 a; and Williams and Tohline 1987 d) have had the same specific angular momentum distribution. The specific angular momentum distribution can be specified as a function of the mass $M_r$ that is enclosed within a cylinder of radius $r$, by the equation

$$j(r) = \frac{J}{M} (1 + q)[1 - (1 - M_r/M)^{1/q}]$$

(7.1)

where $J$ is the total angular momentum, $M$ is the total mass, and $q$ is a parameter. If $q = 3/2$, as it does in the above studies, the specific angular momentum distribution is identical to that of a uniformly rotating, uniformly dense sphere. Therefore, it is the same as the distribution found in any Maclaurin spheroid and belongs to the $n' = 0$ class defined by Ostriker and Mark (1968). Other angular momentum distributions are possible. Hachisu, Tohline, and Eriguchi (1987, 1988) have argued that $\beta_d \approx 0.27$ for any centrally condensed polytrope, but this has not been proven by evolving any of the other axisymmetric equilibria for a $q \neq 3/2$ angular momentum distribution. Hachisu (1986 a, 1986 b), using his very versatile hydrostatic code, has provided me with nine axisymmetric equilibria with $q = 1$. Hachisu's models vary in compressibility, $n$, and $\beta$. For the die-hard fans of this instability's leading to binary fission someway, somehow, there is still an outside chance. Regardless of what happens it should be interesting.
7.2 **WHITE DWARF EQUATIONS OF STATE**

The entire universe is not composed of polytropes! Hachisu (1986 a) can generate equilibria that are appropriate for white dwarf equations of state, too. He has not given me any of these equilibria yet, but if I can properly husband the models he has given me, this too is an interesting field to investigate with a 3-D hydrodynamic code like mine.

7.3 **GRAY RADIATION TRANSPORT**

Boss (1983, 1984, 1986 a, 1986 b) has included gray radiation transport in the Eddington approximation in his 3-D hydrocode. This leads to heating and cooling terms in the energy equation. Also, the equation of state is no longer barotropic. One of the reasons I am excited about working with Alan Boss at DTM is because of this work. I plan to include radiation transport in my future work. Tohline has discovered what is probably a superior way to solve the energy equation (as was discussed in Chapter 2) since the adiabatic, reversible energy transport can be separated from the nonadiabatic irreversible energy sources. These two ideas need to be combined and tested on some real problems.

7.4 **CARTESEAN HYDRODYNAMICS, USING HW3CRT TO SOLVE $\nabla^2 \Phi = 4\pi G\rho$**

In some ways, a hydrodynamic code in Cartesian coordinates in 3-D would be very interesting. One of the reasons this has not been done so far is that if rotation is present, even an axisymmetric model is not guaranteed to conserve angular momentum. Another reason is the absence of a potential solver to use on Poisson's equation to solve for self-gravity. The last problem is removed by
the package of subroutines in the collection HW3CRT in FISHPACK available from NETLIB @ ANL-MCS.arpa over the network. HW3CRT uses a standard Helmholtz seven-point approximation in Cartesian coordinates \((x, y, z)\) on a centered finite difference grid. One glance at the documentation in the comments at the beginning of the program and it is obvious that this package can be turned into a Cartesian 3-D Poisson solver.

Cartesian hydrodynamics has the advantage that the grid resolution is the same all across the model. In a cylindrical or spherical grid, the small zone size in the center reduces the time step because the grid size enters in the numerator of the Courant condition. Cartesian grids have no small zones. Another advantage of Cartesian hydrodynamics is that it is easy to program.

7.5 Adding More Source Terms, Putting in More Physics

The effects of magnetic fields could be added to the 3-D Eulerian hydrocode. There is good evidence that molecular clouds in the lowest density region are kept from collapse for a while by their magnetic fields. The equation of motion has the added source term:

\[
\frac{1}{4\pi} (\mathbf{B} \cdot \nabla) \mathbf{B} - \frac{1}{8\pi} \nabla B^2,
\]

(7.2)

where \(\mathbf{B}\) is the magnetic field.

Many of the clouds are known to be in turbulent motion with small-scale random motion. One method of modeling this small-scale random motion is by using turbulent viscosity; see §2.9 and Marcus (1986). The turbulent motion is
supersonic, but sub-Alfvénic. If there were no magnetic fields, then the supersonic turbulence would quickly die away. So besides using turbulent viscosity to model the small-scale random motion, magnetic fields must be included as well.
REFERENCES


CDC 1981, *CDC Cyber 200 Fortran Version 3* (Sunnyvale, Control Data Corporation).


866.


Walter, F.M. 1987, The Naked T Tauri Stars: the Low-Mass Pre–Main Se-


Winkler, K.-H.A. and Norman, M.L. 1986 a, WH80s: Numerical Radiation


APPENDIX A

The article, "Linear and Nonlinear Dynamic Instability of Rotating Polytropes," is reprinted from The Astrophysical Journal, volume 315, pages 594-601, April 15, 1987. This article comprises Appendix A, pages 97-104 of this dissertation, and is included in the pocket of the binding. The letter from The Astrophysical Journal granting permission to duplicate the article in this dissertation follows:

THE ASTROPHYSICAL JOURNAL

Hilmut A. Abt, Managing Editor
Kitt Peak National Observatory
Box 26732
Tucson, Arizona 85726-6732
602-323-9215

A. Dallaglio, Letters Editor
Center for Astrophysics
40 Garden Street
Cambridge, Massachusetts 02138
617-495-4477

April 10, 1987

Dr. Harold Alden Williams
Department of Physics & Astronomy
Louisiana State University
Baton Rouge, LA 70803-4001

Dear Dr. Williams:

Thank you for your letter of April 7 regarding permission to reprint from a recent paper of your own.

Under our application of the 1978 copyright law, the authors retain the rights to grant reprinting permission. Therefore, you need consult only yourself and Dr. Tohline.

Sincerely,

Helmut A. Abt
Managing Editor

HAA:dw

Published by The University of Chicago Press, 5801 Ellis Avenue, Chicago, Illinois 60637
for THE AMERICAN ASTROPHYSICAL SOCIETY

96
I. INTRODUCTION

The formation of stars and planets from rapidly rotating, self-gravitating, axisymmetric equilibria by the exponential growth of an unstable nonaxisymmetric mode is generally referred to as "fission." The "model space" of these fission events is very large as reviewed by Durisen and Tohline (1985). The present work complements the TDM work exploring the dynamic instability of models having different polytropic indices. The five polytropes considered have $n = 0.8, 1.0, 1.3, 1.5,$ and $1.8$ but the same initial $\beta = 0.310$. The same basic numerical techniques are used as in TDM, although the angular resolution is improved by a factor of 2.

Lucy (1977) and Gingold and Monaghan (1977, 1978, 1979) have studied fission by using a different technique called smoothed particle hydrodynamics. Lucy's (1977) calculation with a $n = 0.5$ and $\beta$ in the range of dynamic instability is the only model exploring a region in the fission model space similar to the present study. The Gingold and Monaghan models either have $\beta < 0.27$, i.e., $\beta$ below dynamic instability, or $\beta > 0.5$, i.e., $\beta$ above gravitational equilibrium.

In § II of this paper is a description of the relevant equations modeled, the numerical techniques, and the initial axisymmetric equilibria. In § III the benefits of using a Fourier analysis of the spatial density distribution in studying evolutions are discussed; in § IV the evolution of the initially axisymmetric equilibria to a decidedly nonaxisymmetric structure is detailed; and in § V conclusions are presented.

II. MODEL DESCRIPTION

The relevant physical equations are four. The equation of motion is

$$\rho \frac{d\mathbf{v}}{dt} = -\rho \nabla \Phi - \nabla P,$$

where $\rho$ is the density, $\mathbf{v}$ is the fluid velocity, $t$ is the time, $\Phi$ is the gravitational potential, and $P$ is the pressure. The equation of continuity is

$$\frac{d\rho}{dt} + \rho \mathbf{v} \cdot \nabla \rho = 0.$$

The barotropic equation of state is

$$P = K \rho, \tag{3}$$

where the constant $K$ specifies the specific entropy of the gas and $\gamma$ is the traditional adiabatic exponent. Throughout the time evolution of each model, $\gamma$ is fixed and related to the chosen polytropic index $n$ of the model's initial structure via the relation $\gamma = 1 + (1/n)$. Finally, Poisson's equation is

$$\nabla^2 \Phi = 4\pi G \rho.$$  

where $G$ is the gravitational constant.

These four nonlinear equations are solved by writing them as finite difference equations on an Eulerian grid in cylindrical coordinates. Henceforth the discrete locations of the grid zones in the coordinates $R, \theta, Z$ will be referred to by their respective integer values $(i, j, k)$ where the total number $n$ of zones used in each direction is $32, 128, 32$, respectively. The time evolution is determined by an explicit time integration of the equations. For more detail on the implementation of this particular hydrocode see Tohline (1978, 1980, 1982).

Two symmetries are used to cut down on computational time. Reflection symmetry is imposed through the equatorial plane $Z = 0$; and a periodic boundary condition is implemented at angle $\theta = \pi$ as well as $\theta = 2\pi$, so that all physical variables are

\begin{align*}
0.35. \text{ where } s & \mathbf{T}', M', \mathbf{T} \text{ is rotational kinetic energy, and } H' \\
\text{Tohline, Durisen. and McCoUough (1985, hereafter TDM) } & \text{explored this problem for models at a fixed polytropic index} \\
\text{growth of an unstable nonaxisymmetric mode is generally} & \text{self-gravitating, axisymmetric equilibria by the exponential} \\
\text{analysis of the spatial density distribution in studying evolu} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{the gravitational potential energy. The initially axisymmetric equilibria are constructed by the Ostriker-Mark, self-consistent field method. The nonaxisymmetric pattern that develops is a two-armed spiral, and} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{is the gravitational potential energy. They found that when} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{used in each direction is (3Z 128, 321 respectively. The time} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{angular resolution is improved by a factor of 2.} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{LSC Observatory Contribution 198.} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{ABSTRACT} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{A three-dimensional hydrodynamic computer program is used to study the growth of nonaxisymmetric} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{structures in rapidly rotating, self-gravitating polytropes. Models with polytropic index } n = 0.8, 1.0, 1.3, 1.5, & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{and 1.8 are studied. The initial ratio } T/|W| \text{ is 0.310 in all models, where } T & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{is rotational kinetic energy and } W \text{ is the gravitational potential energy. The initially axisymmetric equilibria are constructed by the Ostriker-Mark, self-consistent field method. The nonaxisymmetric pattern that develops is a two-armed spiral, and} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{is the gravitational potential energy. They found that when} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{used in each direction is (3Z 128, 321 respectively. The time} & \text{tions are discussed; in § IV the evolution of the initially} \\
\text{angular resolution is improved by a factor of 2.} & \text{axiymmetric equilibria to a decidedly nonaxisymmetric struc} \\
\text{LSC Observatory Contribution 198.} & \text{tions are discussed; in § IV the evolution of the initially} \\

97

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
solving Poisson's equation (see Tohline 1978, 1980). On the defined by Ostriker and Mark. Table 1 summarizes some justification for "n symmetry" is that even modes are expected here because a Fourier transform is already implemented in angle space. This analysis technique is particularly useful in analyzing nonaxisymmetric structures is to Fourier transform the density caused by the mth structure. If one thinks of these structures as contributing to the density then the growth of the mth structure can be studied by plotting c(J,m) versus time.

In order to break the unnatural symmetry of a purely axisymmetric equilibria are calculated by using the consistent field method of Ostriker and Mark (1968). In each model, a specific angular momentum distribution identical to that of a uniformly rotating, uniform density sphere is adopted—that is, each initial model belongs to the $n' = 0$ class defined by Ostriker and Mark. Table 1 summarizes some important properties of the five initial models used in this study. In this table, all dimensional values are given in polytropic units where $G = K = M = 1$, and $M$ is the total mass of the object. The Appendix describes conversion from the "polytropic units" used in this paper to more traditional units.

For each model, $n$ is the polytropic index, $R_p$ is the polar radius, $\rho_r$ is the central density, and $c_p$ is the central angular velocity. The ratio $\rho_r/c_p$, where $\rho_r = 3M/4\pi R_p^3$, is tabulated in order to give a measure of the relative central condensation of each model.

In order to break the unnatural symmetry of a purely axisymmetric model, upon introducing into the three-dimensional hydrocode the density is perturbed once by introducing low-level random fluctuations of the form:

$$\rho(J, L, K) = \rho_a(J, K|1 + \alpha f_{J,L,K}),$$

where $\rho_a$ is the axisymmetric density, $\alpha = 5 \times 10^{-4}$ is the chosen maximum amplitude, and $f_{J,L,K}$ is a random number between $-1.0$ and $1.0$.

### Table 1

<table>
<thead>
<tr>
<th>$n$</th>
<th>$R_p$</th>
<th>$\rho_r$</th>
<th>$c_p$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>2.536</td>
<td>4.445</td>
<td>1.68</td>
<td>10$^{-4}$</td>
</tr>
<tr>
<td>1.0</td>
<td>3.616</td>
<td>4.625</td>
<td>7.99</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>1.5</td>
<td>4.940</td>
<td>4.666</td>
<td>1.79</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>2.0</td>
<td>6.980</td>
<td>4.256</td>
<td>2.52</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>3.0</td>
<td>12.524</td>
<td>5.226</td>
<td>4.256</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>5.0</td>
<td>39.365</td>
<td>5.023</td>
<td>1.920</td>
<td>$10^{-4}$</td>
</tr>
<tr>
<td>8.0</td>
<td>125.990</td>
<td>5.023</td>
<td>1.920</td>
<td>$10^{-4}$</td>
</tr>
</tbody>
</table>

**Note:** Polytropic units

Each of the variables $c_p$ and $\phi_0$ is intrinsically a function of time. Equation (7) can be rewritten as

$$\rho(J, L, K) = \rho_a(J, K|1 + \alpha f_{J,L,K}),$$

where $\rho_a(J, L, K) = c(J, K) \cos (mL\theta - \phi(J, K))$ is the density caused by the mth structure. If one thinks of these structures as contributing to the density then the growth of the mth structure can be studied by plotting $c(J,m)$ versus time.

To see how the mth pattern speed is related to $\phi_0$ consider the axial movement of the maximum of $\rho_a(J, L, K)$. Let $\theta = L\theta$ be the azimuthal angle at which the mth structure has its maximum. In general, $L$ will not have an integer value but must have a value in the range $0 \leq L \leq L_{max}$ at any given time; $L$ by definition is given by the condition $\rho_{max} = 0$. i.e.,

$$mL\theta = \phi(J,m) = \pi n.$$

where $n$ is an integer. The pattern speed $\omega_m$ of the mth structure is then,

$$\omega_m(J,m) = \frac{\partial L}{\partial t} = \frac{1}{m} \frac{\partial \phi(J,m)}{\partial t}.$$

Therefore, the pattern speed $\omega_m$ of any mode $m$ is proportional to the slope of the curve $\phi_0$ versus time, and the proportionality constant is just $m^{-1}$.

How will the growth rate and the pattern speed of the mth pattern be affected by the finite difference technique? Explicit time integration using a first-order finite difference scheme on an Eulerian grid introduces a numerical diffusion that can be represented as a numerical viscosity according to Norman, Wilson, and Barton (1980). Since velocities are small in the $R-$ and $Z$-directions, the model being originally in equilibrium, the only significant viscosity is in terms of angular rotation about the Z-axis. (Velocities do develop radially and in the $Z$-direction as each evolution progresses, but while the developing nonaxisymmetric structure is low in amplitude, they are much smaller than the angular velocity.) A straightforward extension of the Norman, Wilson, and Barton (1980) discussion, as was presented by TDM, shows that the numerical viscosity will cause the numerically calculated change in the density $\delta \rho$ in a time $\delta t$ to differ from the true differential $\partial \rho/\partial t$ by an amount

$$\frac{\delta \rho}{\delta t} \neq \frac{1}{\delta t} \frac{\partial \rho}{\partial t} = \frac{1}{\delta t} \frac{\partial \rho}{\partial \theta} \frac{\partial \theta}{\partial t} = \frac{\partial \rho}{\partial \theta} \frac{\partial \theta}{\partial t} \frac{\partial \theta}{\partial t}.$$

2 Equ. (8a) and (8b) of TDM incorrectly state that the Fourier spectrum extended through $m = 64$ in that paper, $L_{max} = 32$, so the correct statement is that the spectrum extended to $m = 32$.
where \( \omega \) is the angular velocity of the fluid and, as in relations (6a)-(6d), \( \theta = x/L_{\text{rot}} \). Using equation (7) for \( \rho \) on both sides of equation (11) and matching coefficients of like trigonometric form in \( m \) yields

\[
\frac{\partial c_2}{\partial t} - \frac{\partial c_3}{\partial t} = \frac{1}{2} m^2 \omega c_3 \theta .
\]

for the \( \cos(m\theta + \phi_m) \) term, and

\[
\frac{\partial c_2}{\partial t} - \frac{\partial c_3}{\partial t} = 0 .
\]

for the \( \sin(m\theta + \phi_m) \) term. Again, the " \( \delta \) " notation is used to represent numerically determined values while " \( \epsilon \) " denotes a true differential. Equation (12) can be rewritten as

\[
\frac{\delta \ln c_2}{\delta t} = \frac{\delta \ln c_3}{\delta t} + \frac{1}{2} m^2 \omega \delta \theta .
\]

So the numerical growth rates of the \( m \) th structure can be corrected to yield the real growth rates. Notice that numerical viscosity increases with increasing \( m \). As was pointed out in TDM and is again illustrated here in equation (13), the pattern speed of the \( m \) th structure is not affected by numerical viscosity to at least the first order.

IV. EVOLUTIONS

Table 2 shows growth rates and bar mode pattern speeds that were obtained from the five hydrodynamic evolutions and the corresponding quantites obtained from a tensor virial equation (TVE) analysis of each initial axisymmetric model (R. H. Durisen and L. David, private communication). Columns (2) and (5) tabulate the growth rate and pattern speeds of the \( m = 2 \) pattern as determined directly from the numerical simulations. In columns (3), the numerical growth rates are corrected according to equation (14), which compensates for numerical viscosity. Values in columns (3) compare very favorably with the TVE growth rates listed in column (4). The numerically determined pattern speeds (col. (5)] also compare favorably with the TVE values listed in column (6). Figure 1 shows how the growth rates are extracted from the hydrodynamic data. Plotted in this figure for a fixed value of \( \langle U, K \rangle \) is the normalized amplitude

\[
D_m = c_m / \sqrt{\sum c_i^2}
\]

versus time for three different values of \( m \) in the \( n = 1.8 \) evolution. While deviations from axial symmetry are small,

![Figure 1](image-url)

**TABLE 2**

<table>
<thead>
<tr>
<th>( n )</th>
<th>( \frac{\delta \ln c_2}{\delta t} )</th>
<th>( \frac{\delta \ln c_3}{\delta t} )</th>
<th>( \frac{\delta \ln c_3}{\delta t} + 2m \omega \delta \theta )</th>
<th>( \delta \phi_m / \delta t )</th>
<th>( \delta \phi_m / \delta t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>(2)</td>
<td>(3)</td>
<td>(4)</td>
<td>(5)</td>
<td>(6)</td>
</tr>
<tr>
<td>1.8</td>
<td>0.0026</td>
<td>0.0041</td>
<td>0.0064</td>
<td>0.1028</td>
<td>0.0089</td>
</tr>
<tr>
<td>1.5</td>
<td>0.013</td>
<td>0.0194</td>
<td>0.0149</td>
<td>0.0601</td>
<td>0.0437</td>
</tr>
<tr>
<td>1.3</td>
<td>0.031</td>
<td>0.043</td>
<td>0.040</td>
<td>0.1025</td>
<td>0.0994</td>
</tr>
<tr>
<td>1.0</td>
<td>0.099</td>
<td>0.100</td>
<td>0.089</td>
<td>0.242</td>
<td>0.237</td>
</tr>
<tr>
<td>0.8</td>
<td>0.111</td>
<td>0.136</td>
<td>0.133</td>
<td>0.369</td>
<td>0.362</td>
</tr>
</tbody>
</table>

**Note:** Polytropic units

* TVE numbers obtained from R. H. Durisen and L. David (private communication).

The rapid growth of \( D_4 \) and \( D_6 \) at late times as illustrated in Figure 1 was at first very puzzling, especially in the light of equation (14) which clearly shows that these modes should experience strong damping by numerical viscosity. This puzzle is explained by comparing the pattern speeds for the \( m = 2, 4, \) and 6 Fourier components. The time dependence of the phase angles \( \phi_2, \phi_4, \) and \( \phi_6 \) for one particular radial position \( \langle U, K \rangle \) in model \( n = 1.8 \) is shown graphically in Figures 2a, 2b, and 2c, respectively. From this figure alone, it is evident that \( \phi_6 \), moves through 2\( \pi \) radians roughly twice as fast as does \( \phi_4 \), and that \( \phi_2 \) moves through 2\( \pi \) radians roughly 3 times as fast as does \( \phi_4 \). In order to make a more quantitative statement about the relative pattern periods, the ratio of the frequencies \( \delta \phi_4 / \delta t \) and \( \delta \phi_6 / \delta t \) is shown as a function of time in Figures 3a and 3b, respectively. Indeed, the ratios approach the values 2 and 3 at late times. According to equation (10), if the pattern speeds are the same for all Fourier components, then these particular phase angle speeds will be related to each other in ratios of 2 and 3. A careful analysis of the data reveals that not only do the higher modes move together with the \( m = 2 \) pattern, but also their maxima lock in phase with the \( m = 2 \) pattern. It appears, therefore, that the higher order patterns are harmonics of \( m = 2 \) and are not distinct modes; they are to be viewed as contributing to making the bar narrower as higher Fourier components are needed to squeeze the bar in angle space.

In a certain sense, a phase change is occurring, and the time
ROTTING POLYTROPES

Fig. 2. — The phase angle of three different Fourier modes is shown as a function of time for the same radial location in the n = 1.8 model as Fig 1. (a) d1 vs. time; (b) d2 vs. time; (c) d3 vs. time. In the latter part of the evolution, when eigenfrequencies for these Fourier modes become well-defined, curve (a) shows a period roughly twice that of curve (b) and roughly 3 times that of curve (c).

scale for that change to develop is seen. If β = T/|W| is considered a thermodynamic parameter, then, as was seen by TDM, no order develops around the axis when β ≤ 0.25, but when β ≥ 0.30, order develops around the axis of rotation. The symmetry has been spontaneously broken, or at least any symmetry breaking will grow exponentially. In the older language of classical mechanics the equilibrium sequences were said to bifurcate due to a dynamical instability at β = 0.27.

If the nonchaotic parts of Figure 2a are taken and 2π radians are added every time the angle d1 completes a revolution, then δd1/δt is seen to be constant in time. Therefore, the pattern speed of the developing two-armed spiral shows no secular change.

Figures 1 through 3 are shown only for one of the five models (n = 1.8) and at only one radial position in that model. The other four models look qualitatively the same—all the growth rates and pattern speeds change from model to model.

The fact that, for a given model, the growth rate and pattern speed is the same throughout the radial structure of the model is illustrated in Figures 4 and 5. In these three-dimensional graphs, the vertical "amplitude" of the plotted surface is either in D1 or D2. In both figures, the horizontal axes are time and cylindrical radius R. A slice through Figure 4 at the fixed radius 0.40 reproduces the top curve in Figure 1, and a slice through Figure 5 at the same radius reproduces the curve in Figure 2a. The amplitude of the well-developed spiral pattern is always found to be largest at large radii. This is consistent with the behavior of the trial eigenfunction used in the TVE analysis. As was pointed out in TDM, the trial eigenfunction has an Eulerian density perturbation of the form

\[ \delta p = -R \frac{\partial \delta \phi}{\partial R} \]

This function and the radial eigenfunction that develops during the hydrodynamic simulations has its largest amplitude at large radii where the density drops off sharply. (The rise in Fig. 4 at small radii should not be believed. A small nonphysical dimple develops in the density at small radii because of numerical problems associated with the finite difference scheme near the rotation axis.) The analogous Figures 4 and 5 for the other four models look qualitatively the same.

Figure 6 shows the flat, axisymmetric equilibrium for the n = 1.3 model at the beginning of its evolution. Figure 6 actually adequately represents the shape of any of the initial polytropic models with β = 0.310. Figures 7-11 show isodensity surfaces at six different times during the evolution of each of the five models. Although each model develops a two-armed spiral structure, the evolutions clearly look quantitatively different depending upon polytropic index. The n = 1.8
model in Figure 11 shows the narrowest spiral as expected from its greater degree of compressibility. The \( n = 0.8 \) model in Figure 7 shows the broadest spiral as expected from its smaller degree of compressibility. We have produced a color movie of the time evolution of the density contours in the equatorial plane for the two models \( n = 0.8 \) and \( n = 1.8 \). The difference that compressibility makes in the development of the spiral pattern in these two extreme examples is also very evident when viewed in this manner.

V. CONCLUSIONS

TDM used a three-dimensional hydrodynamic computer code to study the dynamical growth of nonaxisymmetric structure in a rotating star. The goal was to gain insight into the behavior of the star's internal structure and how it evolves over time.

Fig. 6—A single isodensity surface contour is shown for the initial axisymmetric \( n = 1.3 \) model. The other four isodensity contours correspond to the final state of the simulation.

Fig. 7—A series of isodensity surface contours for the \( n = 0.8 \) model. Times are indicated in polytropic units. The model develops a trailing spiral pattern, indicating the development of a nonaxisymmetric structure.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
ture in rapidly rotating, \( n = 1.5 \) polytropes. This study is a complementary one to that in the same computer code—but with twice the angular resolution—has been used to study the dynamical growth of nonaxisymmetric structure in rapidly rotating polytropes having five different values of the polytropic index \( \{n = 0.8, 1.0, 1.3, 1.5, 1.8\} \). As in TDM, strong agreement between the results of the numerical simulations and the predictions of linear theory—specifically, predictions made by a tensor virial equation analysis—have been obtained. A \( \delta p / \rho \propto \cos \theta \), "barlike" nonaxisymmetric structure grows exponentially out of low-level random fluctuations in an otherwise axisymmetric model. As in TDM, the nonaxisymmetric structure develops as an eigenmode having a well-defined growth rate, a well-defined pattern speed, and a distinctly spiral character. In all models, the pattern speed matches the TVE predictions to within 15\% (the \( n = 0.8 \) model distinctly spiral character. In all models, the pattern speed matches the TVE predictions to within 15\% (the \( n = 0.8 \) model differs by only 2\%) and, when a standard correction is made to account for the artificial diffusion that is implicit to the low-order numerical scheme, the growth rates observed in all models also match the TVE predictions to within 15\%. The quantitative consistency between the TVE analysis and these numerical simulations lends considerable credence to both techniques as being useful in studying the dynamical stability of rapidly rotating, self-gravitating objects.

In the TVE analysis a pure bar—nested ellipsoids with two equal axes transforming into nested triaxial ellipsoids—is used as the trial eigenfunction. In Chandrasekhar's (1969) classical work these triaxial ellipsoids are Rieman ellipsoids. Spirals are not ellipsoidal figures. In the future, TVE analyses should adopt a more general trial eigenfunction that allows for the development of spiral eigenmodes. TDM found that, for a fixed polytropic index, the spiral pattern that develops spontaneously is more tightly wound for the more rapidly rotating (large \( \Omega \)) models. As a fixed \( \Omega / W \) and for a fixed distribution of angular momentum, the present work shows that the spontaneously developing spiral pattern is more tightly wound in more compressible (higher polytropic index) models. The more compressible models possess initial axisymmetric structures that are more centrally condensed. Therefore, for the chosen distribution of angular momentum \( (n = 0) \), the more compressible models exhibit a much larger degree of differential rotation. This, in itself, may account for the development of a more tightly wound pattern in the models with higher polytropic indices.

Each model evolution presented here has been stopped at the point where the amplitude of the growing spiral eigenmode has become nonlinear—i.e., \( \delta p / \rho \approx 1 \). The primary purpose has been to study the results of such modes in the linear amplitude regime in order to establish a secure tie between the employed numerical technique and linear theory. The hydrodynamic code that has been used integrates the complete nonlinear set of coupled partial differential equations, however, so it is possible to extend each of these evolutions to much longer times. The ultimate goal, of course, is to see whether such an instability can lead to the "fission" of a single rapidly rotating object into a close binary star system. The single published evolution in this ongoing investigation that has been carried to completion (Dursien et al. 1986) was for \( n = 1.5 \) polytrope. Fission into a binary system was not the outcome of that evolution, as the nonlinear amplitude, tightly wound spiral pattern allowed an efficient redistribution of angular momentum to occur. A ring of material was shed from the outer layers of the object in lieu of direct fission. It will be interesting to see whether the completion of an evolution for a model with a stiffer equation of state—and a correspondingly less tightly wound spiral eigenmode—can also avoid direct fission.

Numerous discussions with N. Lebovitz and R. H. Dursien throughout our efforts on this problem have been extremely beneficial. We are grateful to L. David and R. H. Dursien for providing the TVE numbers in columns (4| and (6) of Table 2. We also appreciate the very valuable assistance that F. Shu provided in our attempt to understand the time evolutionary behavior of the high-order Fourier modes. This work is supported, in part, by the National Science Foundation through grant AST 85-01842. A portion of this work was carried out while one of us (J. E. T.) was participating in the Star Formation Program at the Institute for Theoretical Physics, which is supported in part by the National Science Foundation under grant PHY 82-17853, supplemented in part by funds from the National Aeronautics and Space Administration.

**APPENDIX**

**POLYTROPIC UNITS TO OTHER UNITS**

In Tables 1 and 2, dimensional quantities are given in "polytropic units" defined such that \( G = K = M = 1 \). All quantities with natural units of inverse time, such as \( \omega_b \) in Table 1 or columns (12)-(6) of Table 2, can be put into the units used by Ostriker and Bodenheimer (1973) by simply multiplying the tabulated values by \( (4\pi\rho)^{-1/2} \). TDM provided such a conversion directly in their Tables 2, 3, and 4. This physically motivated choice of time units proves to be a very good one because, in these units, the eigenfrequencies for the \( m = 2 \) mode have values that are nearly independent of the polytropic index. In fact, in the time units used by Ostriker and Bodenheimer, the growth rates and pattern speeds for models having \( r = 0.310 \) vary at most by \( 17\% \) from one another as the polytropic index is changed from a value \( n = 0.8 \) to \( n = 1.8 \).

A mass \( M \) and polytropic constant \( K \) must be specified before the dimensional quantities listed in Tables 1 and 2 can be converted into more common units. Using equation (2), the common dimensions of the polytropic constant \( K \) are \( m^{n-1} r^{2/3} d^{-2} \), where \( m \), and \( d \) denote mass, time, and distance. The gravitational constant \( G \) has common dimensions of \( m^{-1} r^{2} d^{2} \). Therefore, in order to convert any quantity \( q_{\text{pc}} \) reported here in polytropic units into its associated quantity \( q \) whose common units are \( m^{n-1} r^{2} d^{2} \), one simply uses the expression

\[
q = q_{\text{pc}} (G^* M^* K^*) ^ n .
\]

(A1)
where, for a given polytropic index $n \neq 3$,

$$\alpha = \frac{3}{(n - 3)} \left[ \sigma + \frac{1}{2} \right].$$

$$\psi = \frac{3}{(n - 3)} \left[ \ln \left( \frac{R}{R_o} \right) + \sigma \right].$$

$$\Theta = \frac{n}{(n - 3)} \left[ \frac{3}{2} \sigma \right].$$

Specifically, then, to convert any radius $R_{m}$ reported in Table 1 to a radius having the common dimension $d$,

$$R(d) = R_{m} \left( \frac{G M_{c}^{2} K^{n-d} \sigma^{1} \Theta^{1}}{R_{m}^{3}} \right),$$

(A5)

to convert a density $\rho_{m}$ reported in Table 1 to a density having the common dimensions $m d^{-3}$,

$$\rho(d^{-3}) = \rho_{m} \left( \frac{G M_{c}^{2} K^{n-d} \sigma^{1} \Theta^{1}}{R_{m}^{3}} \right),$$

(A6)

and to convert any frequency quantities $\omega_{m}$ given in Tables 1 and 2 (like a growth rate or a pattern speed) to frequencies $\omega$ having the common dimensions $t^{-1}$,

$$\omega(t^{-1}) = \omega_{m} \left( \frac{G M_{c}^{2} K^{n-d} \sigma^{1} \Theta^{1}}{R_{m}^{3}} \right),$$

(A7)

Often, a physical problem does not readily provide a value for the specific entropy $K$ since its common dimensions can involve $m$ and $d$ raised to fractional powers. Instead of $M$ and $K$, then, a choice of $M$ and $R_{m}$ or a choice of $M$ and $\rho$ is often preferable. Given $M$ and any common scale length $R$, the alternate form for equation (A6) is

$$\rho(d^{-3}) = \rho_{m} \left( \frac{M}{R^{3}} \right),$$

(A8)

and the alternate form for equation (A7) is

$$\omega(t^{-1}) = \omega_{m} \left( \frac{G M}{R^{3}} \right).$$

(A9)

Given $M$ and a density $\rho$, the alternate form for equation (A8) is

$$R(d) = R_{m} \left( \frac{M \rho_{m}}{\rho} \right)^{1/3},$$

(A10)

and the alternate form for (A7) is

$$\omega(t^{-1}) = \omega_{m} \left( \frac{G M}{\rho} \right)^{1/3}.$$

(A11)

Relations (A8)-(A11) do not depend explicitly on the polytropic index $n$, but a definite dependence on $n$ still persists implicitly through the quantities $R_{m}$ and $\rho_{m}$.

Two brief examples serve to illustrate the conversion from polytropic units to common units. For an $n = 1.5$ polytrope having a $T/W^{1/3} = 0.310$, a mass $M = 1 M_{Q}$, and an equatorial radius $R_{m} = 1 R_{Q}$, relation (A5) combined with the $R_{m}$ (in polytropic units) from Table 1 indicates that, in inverse seconds, $\omega(t^{-1}) = 2.7 \times 10^{-5} \omega_{m}$. Therefore, the $e$-folding time for the dynamically growing $m = 2$ mode in this solar-type object is $t_{2} = [7 \ln (e^{2})]^{-1} = 0.53$ hr. For a much more distended $1 M_{O}$ protostellar object, say one with $R_{m} = 40 A.U.$ so that it encompasses the entire volume of the current solar system, the $e$-folding time for this mode is $t_{2} = 48$ yr.

REFERENCES


JOLY E. TOHLINE and HAROLD A. WILLIAMS: Louisiana State University, Baton Rouge, LA 70803-4001
APPENDIX B
CIRCUMSTELLAR RING FORMATION
IN RAPIDLY ROTATING PROTOSTARS

HAROLD A. WILLIAMS AND JOEL E. TOHLINE

Department of Physics and Astronomy, Louisiana State University

Submitted to The Astrophysical Journal

ABSTRACT

A three-dimensional hydrodynamic computer program is used to study rapidly rotating, self-gravitating polytropes with polytropic indices \( n = 0.8 \) and \( n = 1.8 \). The initial ratio of the rotational to gravitational potential energy is \( \beta_i \equiv E_{\text{rot}}/|E_{\text{grav}}| = 0.310 \) in both models. Both models, initially axisymmetric, are dynamically unstable toward the development of a nonaxisymmetric, two-armed spiral structure, and previous work has demonstrated the behavior of the instability in the linear amplitude regime. Here the models are shown as they evolve to extremely nonlinear amplitudes: the end result in both cases is a type of fission. The models shed a fraction of their mass and angular momentum, producing a ring which surrounds a more centrally condensed object. At the end of the evolution \( \beta \) has been reduced below \( \beta_d = 0.27 \). The central object is a triaxial figure that is rotating about its shortest axis. At the end of each simulation, the central object appears to have entered a steady phase of radial, adiabatic pulsation. The low-mass, roughly axisymmetric ring appears to be dynamically stable but presumably is capable of condensing into planets over a much longer time scale.
I. INTRODUCTION

Rotation is certainly one of the most important factors to be dealt with when attempting to model or explain the process by which protostars become stars. A typical protostellar gas cloud must shed a considerable amount of angular momentum before it can become sufficiently dense to be called a star (Bodenheimer 1983). Magnetic braking, viscous torques, and gravitational torques have been suggested as methods for angular momentum redistribution. Cassen, Shu, and Terebey (1985) have reviewed protostellar disks and star formation and have given a good qualitative description of all three of these methods.

In a fully or partially ionized gas cloud, magnetic braking can certainly be very important. Mouschovias (1977, 1978, 1981), for example, has argued persuasively that this process removes the bulk of the angular momentum from a typical protostellar cloud. When the cloud becomes sufficiently dense, however, the fractional ionization of the gas will become very small, the magnetic field will diffuse fairly rapidly out of the cloud, and magnetic braking will become unimportant. Even if magnetic braking successfully removes several orders of magnitude of the initial total angular momentum from gas clouds during their early phase of contraction, many of the resulting protostars will still be rotating very rapidly. In our present work, we will not investigate the effects of magnetic fields, so our models most accurately represent what is likely to occur during the denser phases of protostellar contraction. We will also not attempt to model the effects of viscous torques, since they cannot easily be modeled from first principles in a rigorous fashion. In the rapidly rotating protostellar models presented here, gravitational torques (resulting from dynamically growing two-armed spi-
ral structures) and their global effects evolved over time are rigorously calculated from first principles as a method of angular momentum redistribution.

Redistribution of angular momentum when it leads to redistribution of mass can cause an initially axisymmetric gas cloud to undergo fission. The fission that the gas cloud undergoes will not necessarily lead to the formation of a binary star system, although this has been the fond hope of many researchers who have explored this problem. It proves instructive, when discussing the subject of fission, to characterize the relative importance of rotation in a protostellar cloud by the dimensionless parameter $\beta \equiv E_{\text{rot}}/|E_{\text{gra}}|$, where $E_{\text{rot}}$ is the rotational kinetic energy and $E_{\text{gra}}$ is gravitational potential energy. Tassoul (1978) points out the usefulness of this parameter in studying gravitational/rotational instabilities. There is no virial equilibrium at $\beta > 0.5$, for example. Furthermore it has been shown that the first nonaxisymmetric dynamical instability for uniformly rotating, incompressible objects occurs at a $\beta = \beta_d \equiv 0.27$ (Lyttleton 1953, Chandrasekhar 1969, and Tassoul 1978). This particular dynamical instability allows a structural deformation to grow exponentially in time. For incompressible systems, the instability that occurs at $\beta = \beta_d$ gives rise to a deformation that is ellipsoidal in shape, so it is often referred to as a bar-mode instability. It is the analysis of this classical bar-mode instability that we are studying here in an effort to better understand the evolution of rapidly rotating protostars.

In order to fairly represent the nonaxisymmetric distortions that are likely to arise dynamically when a gaseous protostar encounters the dynamical bar-mode instability, we must model the evolution hydrodynamically with full three-dimensional (3D) generality and must use something other than an incompress-
ible equation of state. Because the 3D simulation itself introduces a significant degree of complication into the problem, we have chosen to use a fairly simple representation of the equation of state; a polytropic relation characterized by an index \( n \neq 0 \) has been chosen (see §IIb). A linearized tensor virial equation analysis of the properties of rapidly rotating, axisymmetric polytropes has revealed that a bar-mode instability is encountered in these compressible systems at very nearly the same point that it occurs in incompressible objects; that is, at \( \beta \approx \beta_d \) (eg. Tassoul and Ostriker 1970; Ostriker and Bodenheimer 1973; see Durisen and Tohline 1985 for a review). Using 3D hydrodynamic computer codes, several groups have recently confirmed the predictions made by linear theory. Polytropes having \( n = 1.5 \) and \( n = 0.5 \) have been shown to be dynamically unstable toward the development of nonlinear, nonaxisymmetric structure if, initially, \( \beta > 0.30 \) (Lucy 1977; Gingold and Monaghan 1978, 1979; Durisen and Tohline 1980; Tohline, Durisen, and McCollough 1985; Durisen, Gingold, Tohline, and Boss 1986; Durisen and Gingold 1986). Tohline, Durisen, and McCollough (1985) have, furthermore, carefully determined that in \( n = 1.5 \) polytropes having \( \beta = 0.30, 0.33, \) and 0.35, the naturally developing nonaxisymmetric eigenmode exhibits both a growth rate and a pattern speed that is in quantitative agreement with the predictions of linear theory. Most recently, Williams and Tohline (1987, hereafter WT) have reported that in the linear growth regime five models having identical initial values of \( \beta (\beta_i = 0.310) \), but having different degrees of compressibility \( (n = 0.8, 1.0, 1.3, 1.5, \) and 1.8), also exhibit growth rates and pattern speeds in quantitative agreement with the predictions of linear theory. Somewhat surprisingly, however, in all of the studies of centrally condensed, differentially
rotating objects, the bar-like mode that grew exponentially was actually found to be a two-armed spiral mode.

In the present work, two of the five models whose early linear amplitude regime evolution was studied by WT are carried to extremely nonlinear amplitudes. In both cases, following a short period of undergoing extreme nonaxisymmetric distortions, the models settle into a state composed of a fairly axisymmetric ring surrounding a much more centrally condensed triaxial object. In the final state, the global $\beta$ has been reduced below $\beta_d$. The formation of a similar circumstellar ring structure has been seen before in other fission studies (Durisen and Tohline 1985; Durisen, Gingold, Tohline, and Boss 1986; see §IV for a discussion), but all previous studies have used models with $\beta_i > 0.310$. It is reasonable to expect that a slowly contracting protostar will never evolve to a configuration in which $\beta$ is substantially larger than $\beta_d$ if it undergoes a dynamical instability at or near $\beta_d$ that results in redistribution of angular momentum and a corresponding reduction in the value of $\beta$. The present study which examines models having $\beta_i = 0.310$ therefore brings us closer to an understanding of what happens in systems that are only marginally unstable.

This study also examines the differences that can arise in “fission evolutions” when the degree of compressibility of the gas is changed. The two polytropes with the same $\beta_i$ but different compressibility show marked differences in where the ring forms and in what fraction of the object mass goes into the ring.

In §II of this paper is a description of the initial model, the modeled physics, and the solution method in the hydrocode; in §III an extensive discussion of the dynamical evolution of the two models is presented; and in §IV our general
conclusions are summarized.

II. DEFINITION OF THE PROBLEM

a) Initial Models

When performing a numerical hydrodynamic simulation of any type, it is important to select one's starting configuration—eg., the initial density distribution and velocity flow field—with care because the outcome of the simulation will often be extremely dependent on initial conditions. This is inevitable because, by their very nature, most nonlinear hydrodynamic problems are initial-value problems. When the outcome is sensitive to initial conditions, then the chosen starting configuration must be a physically realistic one in order for the simulation to be categorized as anything more than just an abstract numerical experiment. Our current study focuses on a particular set of models whose properties are not only astrophysically interesting, but whose evolutions are robust—that is, the results of each evolution prove to be insensitive to the exact spectrum of fluctuations that are introduced into the starting configuration.

Specifically, our initial models are rapidly rotating polytropes whose axisymmetric structures are in hydrostatic equilibrium. By starting with a self-gravitating model that is at least in radial equilibrium, we are assured that the model will not collapse or diffuse away in one free-fall time, $\tau_{ff} \equiv 1/(4\pi G \rho)^{1/2}$. Furthermore, with a well-defined equilibrium state the region in the model space is defined and the experiment becomes reproducible. This allows comparisons to be made between our results and the results of other groups using different numerical methods (eg., Durisen, Gingold, Tohline, and Boss 1986). Our study
also provides an excellent overlap with linear theory. As we mentioned in §1, for our chosen equilibrium states the tensor virial equation analysis has something definite to say about which, if any, nonaxisymmetric distortions will grow, at what rate the distortions will grow, and what eigenfrequencies or pattern speeds the distortions will exhibit. We have chosen to evolve models with a fairly large initial value of $\beta$ ($\beta > \beta_d$) so that the nonlinear outcome of a well-known dynamic instability can be investigated. The robustness of our simulations results from the fact that the instability being studied is a dynamical one. The fastest growing nonaxisymmetric eigenmode appears spontaneously out of "noise." The outcome of each 3-D evolution should justifiably be independent of the spectrum of the initial distortion, $\delta \rho / \rho$, that is introduced into the well-defined axisymmetric state.

For the present study, initial axisymmetric equilibria are calculated by using the self-consistent field method of Ostriker and Mark (1968). The chosen specific angular momentum distribution, specified as a function of the mass $M_r$ that is enclosed within a cylinder of radius $r$, is

$$j(r) = \frac{J}{M}(1 + q)[1 - (1 - M_r/M)^{1/q}],$$

where $J$ is the total angular momentum, $M$ is the total mass, and for the current study $q = 3/2$. This gives a specific angular momentum distribution identical to that of a uniformly rotating, uniform density sphere. Therefore it is the same as the distribution found in any Maclaurin spheroid and belongs to the $n' = 0$ class defined by Ostriker and Mark (1968); but the models studied here are not constructed using a polytropic index $n = 0$, so they are both centrally condensed and
differentially rotating. We have chosen this $j(r)$ relation instead of, for example, constant angular velocity because axisymmetric equilibria having $\beta > \beta_d$ cannot be constructed for compressible polytropes having constant angular velocity. As far as we are aware, all hydrodynamic simulations done with axisymmetric equilibrium with a $\beta$ in this region of the model space have been done with $q = 3/2$. Other angular momentum distributions are possible, but Hachisu, Tohline, and Eriguchi (1987 and 1988) have argued that $\beta_d \approx 0.27$ for any centrally condensed polytrope.

Table 1 summarizes some important properties of the chosen initial axisymmetric models. In Table 1, $n$ is the polytropic index, $R_{eq}$ is the equatorial radius, $R_p$ is the polar radius, $\rho_c$ is the central density, $\bar{\rho} \equiv 3M/(4\pi R_{eq}^2 R_p)$ is a measure of the mean density, and $\omega_c$ is the initial central angular velocity. The units used in Table 1 and throughout the calculation are polytropic units, $G = K = M = 1$, where $G$ is the gravitational constant, and $K$ specifies the specific entropy of the system (see the barotropic equation of state below). Although our primary focus will be on the application of these models to protostars, polytropic units are used throughout the study so that the results can be scaled to other interesting astrophysical systems (see §IV). To convert polytropic units to cgs units or to any other system, see the appendix in WT. If, for example, we choose to associate our $n = 1.8$ model with a $1M_\odot$ protostar having an equatorial radius $R_{eq} = 100$ AU—i.e., having a size comparable to our current Solar system—then our initial model has a central rotation period of 270 years, an average density $\bar{\rho} = 7.9 \times 10^{-13}$ g cm$^{-3}$, a central density $\rho_c = 6.9 \times 10^{-12}$ g cm$^{-3}$, an average hydrogen number density of $\bar{n}_H = 2.3 \times 10^{11}$ cm$^{-3}$, a central hydro-
gen number density of \( n_{e\text{H}} = 2.0 \times 10^{12} \text{ cm}^{-3} \) for \( \mu = 2 \), an average pressure \( \bar{P} = 7.0 \times 10^{-3} \text{dynes cm}^{-2} \), a central pressure \( P_c = 2.05 \times 10^{-1} \text{dynes cm}^{-2} \), an average temperature \( \bar{T} = 220K \), and a central temperature \( T_c = 743K \).

The two initial models in this study are not actually the purely axisymmetric models generated by the Ostriker-Mark code. Instead, they are two of the end states of the dynamical evolutions reported in WT. In WT, both axisymmetric equilibrium models identified in Table 1 were introduced into the computational grid of the 3-D hydrodynamic code and, at each zone in the 3-D grid, the density structure was perturbed by a random density fluctuation of mean amplitude \( \delta \rho / \rho \sim 2.5 \times 10^{-4} \). In this way, at the outset of both simulations no particular distortional mode was selected preferentially. During the first few rotation periods of each evolution, a two-armed spiral mode developed spontaneously out of the initial random fluctuations and its amplitude was observed to grow exponentially in time, as expected for an eigenmode of a dynamically unstable system. In addition, as the linear amplitude (\( \delta \rho / \rho << 1 \)) spiral modes developed, they were observed to move azimuthally through the nearly axisymmetric models with a well-defined pattern speed \( \Omega_2|_{linear} \). In both the \( n = 1.8 \) model and the \( n = 0.8 \) model, the pattern speed and exponential growth rate of the two-armed spiral mode were quantitatively measured and shown to compare very favorably with the linear theory predictions obtained from a tensor virial analysis of the initial axisymmetric models (see WT for details). The pattern speed \( \Omega_2|_{linear} \) identified in WT is tabulated for both models in column 2 of Table 2. Once it became well-defined in both models, the spiral mode exhibited a constant value of \( \Omega_2|_{linear} \) until the amplitude of the spiral became nonlinear (i.e., \( \delta \rho / \rho \sim 1 \)). In terms of
its central rotation period $t_c \equiv 2\pi/\omega_c$, a nonlinear amplitude was reached in the $n = 1.8$ polytrope at a time $t \sim 9.5t_c$. A nonlinear amplitude was reached in the $n = 0.8$ polytrope at a time $t \sim 6.5t_c$. It is at a time just before the nonaxisymmetric distortion reaches a nonlinear amplitude in both of these models that we begin the current study. Hence, both of our models are initially axisymmetric in their outward global appearance, but they both already have embedded in their structure a well-defined, exponentially growing, two-armed spiral perturbation. In each case, the perturbation is exactly the fastest growing natural eigenmode of the system.

In §I we mentioned the self-consistent modeling of angular momentum transport via gravitational torques. Because our initial models each possess a well-defined two-armed spiral pattern which moves through the fluid with a well-defined pattern speed $\Omega_2|_{\text{linear}}$, the conditions are right for this type of transport to occur. With this in mind, it is important that we identify the radial location of a few classical resonance points in our initial models—namely, corotation and the inner and outer Lindblad resonances. Figure 1 provides a plot of $\omega_i/\omega_c$ versus $r$ (solid line) and $(\omega_i + \kappa_i/2)/\omega_c$ versus $r$ (dashed line) for the $n = 1.8$ polytrope, where $\omega_i$ is the initial angular velocity of the gas, $r$ is the cylindrical radius, and

$$\kappa_i \equiv 2\omega_i[1 + \frac{r}{2\omega_i} \frac{d\omega_i}{dr}]^{1/2}$$

is the epicyclic frequency. Figure 2 provides a plot of the analogous information for the $n = 0.8$ polytrope. In each figure, Keplerian angular velocities are plotted for the region $r/R_{eq} > 1$. In the region $r/R_{eq} > 1$ there is initially no mass, but the Keplerian angular velocity identifies the circular velocity with which an
infinitesimally small mass would orbit the cloud. The horizontal chain-dashed line in both figures also identifies $\Omega_2|_{linear}$ in order to show where corotation and the Lindblad resonances fall in each model.

An inner Lindblad resonance does not exist in either model because $\omega_i - \kappa_i/2$ is nearly zero and, hence, less than $\Omega_2|_{linear}$ throughout both models. It is also clear from Figures 1 and 2 that the outer Lindblad resonance resides outside of the initial mass distribution in both models. The pattern speed of the two-armed spiral is such that corotation also lies outside of the $n = 0.8$ polytrope; but in the $n = 1.8$ polytrope, corotation lies inside the cloud at a radius $0.88R_{eq}$.

b) Physics and Solution Method

The physical equations solved by the 3-D hydrodynamic code are the equation of continuity, the equation of motion, a barotropic equation of state, and Poisson's equation for gravity. The four principal dynamical variables are: the mass density $\rho$, the angular momentum density $A$, the cylindrical radial momentum density $S$, and the z-direction momentum density $T$. The continuity equation,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \nu) = 0,$$

and the three components of the equation of motion,

$$\frac{\partial S}{\partial t} + \nabla \cdot (S \nu) = -\rho \frac{\partial \Phi}{\partial r} - \frac{\partial P}{\partial r} + \frac{A^2}{\rho r^3},$$

$$\frac{\partial T}{\partial t} + \nabla \cdot (T \nu) = -\rho \frac{\partial \Phi}{\partial z} - \frac{\partial P}{\partial z}.$$
Fig. 1.—For the $n = 1.8$ polytrope, normalized angular velocities in the initial model are plotted versus the normalized radius $r/R_{eq}$: $\omega_i/\omega_c$, solid line; $(\omega_i + \kappa_i/2)/\omega_c$, short-dashed line. The normalized two-armed spiral pattern speed $\Omega_2/\omega_c|_{linear}$ is also identified by the horizontal chain-dashed line.
Fig. 2.—For the $n = 0.8$ polytrope, normalized angular velocities in the initial model are plotted versus the normalized radius $r/R_{eq}$: $\omega_i/\omega_c$, solid line; $(\omega_i + \kappa_i/2)/\omega_c$, short-dashed line. The normalized two-armed spiral pattern speed $\Omega_2/\omega_c \mid_{linear}$ is also identified by the horizontal chain-dashed line.
\[ \frac{\partial A}{\partial t} + \nabla \cdot (Av) = -\rho \frac{\partial \Phi}{\partial \theta} - \frac{\partial P}{\partial \theta}, \]

where \( v \equiv (v_r, v_z, v_\theta), \ v_r \equiv S/\rho, \ v_z \equiv T/\rho, \ v_\theta \equiv A/(\rho \rho), \) form the four coupled nonlinear PDE's which are used to update the four principle dynamical variables in time. The hydrodynamic equations are solved explicitly in time on an Eulerian grid in cylindrical coordinates. A finite-difference, donor cell technique is used, so the scheme is accurate only to first order in the spatial differences.

The gas pressure \( P \) is a slave of \( \rho \) through the barotropic equation of state

\[ P = K \rho^{1+1/n}, \]

where \( K \) specifies the specific entropy and \( n \) is the polytropic index. The models are initially homentropic and are evolved adiabatically, so a separate equation to follow the thermal energy of the gas is not required. The gravitational potential \( \Phi \) is a slave of \( \rho \) through the Poisson equation

\[ \nabla^2 \Phi = 4\pi G \rho. \]

The Poisson equation is solved by a partial spectral method: the variables \( \rho \) and \( \Phi \) are Fourier transformed in the azimuthal direction and rewritten as a set of 2-D Helmholtz equations which, in turn, are solved directly by Buneman cyclic reduction.

The grid resolution in \( (r, z, \theta) \) is \((64,32,128)\). In both models the grid extends to \( 2.56R_\text{eq} \) in the \( r \)-direction, allowing for radial expansion of the gas cloud. In the \( z \)-direction, the grid extends to \( 1.44R_p \) in the \( n = 1.8 \) polytrope and to \( 1.54R_p \) in the \( n = 0.8 \) polytrope. The axis of rotation is the \( z \)-axis. Symmetry in the
equatorial plane is assumed, so only 16 grid zones in the $z$ direction are actually needed in the computational mesh; and \( \pi \) symmetry in the $\theta$ coordinate is assumed, so only 64 azimuthal grid zones are actually calculated. (In the angular coordinate, \( \pi \) symmetry means that the dynamical variables have a periodic structure such that they are the same in the range \( 0 \leq \theta < \pi \) as they are over the range \( \pi \leq \theta < 2\pi \).) The boundary condition used along the outer cylindrical surface of the grid is "free"; that is, mass is allowed to flow out of the computational grid. The amount of mass and angular momentum lost at the edge of the computational grid is continuously tabulated, but the lost material itself is ignored once it leaves the grid. Initially, the model is well inside the computational grid, so matter does not simply trickle away at the free boundary. During an evolution, however, mass and angular momentum can be lost from the system if, for example, significant radial expansion of the gas cloud occurs.

For additional details on the hydrocode and its implementation see Tohline (1978, 1980, 1982). All calculations were done with this well-tested code, which (in the linear growth regime) has produced results in close agreement with the tensor virial equation analysis (see WT and Tohline, Durisen, and McCollough 1985).
III. EVOLUTIONS

Both models were evolved for many rotation periods beyond the WT study. The major dynamical events during each evolution are depicted in Figures 3 and 4. Figure 3 shows isodensity surfaces at \( \rho/\rho_c = 5 \times 10^{-3} \) for the \( n = 1.8 \) polytrope as it develops in time. The time units written on the face of the figure are in terms of the initial central rotation period \( t_c \). Figure 4 shows isodensity surfaces at \( \rho/\rho_c = 1 \times 10^{-3} \) for the \( n = 0.8 \) polytrope as it develops in time. Figures 3 and 4 illustrate well the primary global features that have arisen during the evolution of the models. First, the "bar-like" natural eigenmode of the system grows to nonlinear amplitude and appears as an obvious two-armed spiral structure (panel b). The spiral elongates (panel c), then wraps up on itself (panel d), forming what ultimately becomes an equatorial, "circumstellar" ring (panel e). For a short time the ring remains partially attached to the central object (panel e), which itself has contracted somewhat and has become more dense. The bridge between the ring and central object ultimately breaks, at least as viewed at certain isodensity levels. The final structure (panel f) is a central triaxial protostar surrounded by a fairly axisymmetric ring. Notice that, in each evolution during the spiral-arm elongation phase, the cloud expands outside its initial radius, but that in the \( n = 1.8 \) case the ring, as well as the central object, contracts within the initial radius \( R_{eq} \). The spiral-arm elongation phase also identifies the interval during which a certain fraction of the mass and angular momentum is ejected from the cloud (see below). The velocity of the matter when it flows off the computational grid is generally less than the escape velocity from the cloud, but not by a large factor. Panel f in both Figures 3 and 4 clearly shows the final circumstellar
ring structure. The evolution from panel e to panel f is very slight. The ring is essentially stable over this entire period.

Looking at a few of the isodensity surfaces does not completely convey what happens during an individual evolution, but looking at several hundred and selecting representatives of different shapes in time convey at least a general picture of what is going on. Only a movie of families of isodensity surfaces in time would really convey the way the mass distribution is changing. Also, in order to understand the density structure as it changes in time, you have to look at more than just one isodensity level at each point in time. Figure 5 shows isodensity surfaces for the $n = 1.8$ polytrope at a fixed point in time ($12.2t_c$) during the spiral arm elongation phase, but at different density levels, where the specified density level in each case is just $\rho/\rho_c$. Similarly, Figure 6 shows four different isodensity surfaces for the $n = 1.8$ polytrope at a fixed point in time, but in this case the structure is shown at the end of the modeled evolution ($19.4t_c$). In Figure 6 the central object clearly appears triaxial, and the surrounding ring is recognized to be the highest density region at the inner edge of a much more distended, low density disk.

Beyond examining the global geometric distortions of the models, we have carefully followed the time-variation of a number of important physical variables. Examining the time-variation of the spiral-mode pattern speed is very instructive. Figure 7a shows the pattern speed $\Omega_2$ versus time in the equatorial plane of the $n = 1.8$ polytrope, at radius $r/R_{eq} = 0.64$. (A plot of $\Omega_2$ versus time at all other radial locations looks very similar to Figure 7a so, throughout the evolution, the pattern frequency proves to be a globally important parameter.) The curve in
Fig. 3.—For the $n = 1.8$ polytrope, a series in time of isodensity surfaces where $\rho/\rho_c = 5 \times 10^{-3}$. Times are indicated in terms of central rotation periods $t_c$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Fig. 4.—For the $n = 0.8$ polytrope, a series in time of isodensity surfaces where

$\rho/\rho_c = 1 \times 10^{-3}$. Times are indicated in terms of central rotation periods $t_c$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Fig. 5.—Four different isodensity surfaces are shown at a fixed point in time, $t = 12.2t_c$, during the evolution of the $n = 1.8$ polytrope. The value of the density at which each surface is drawn is indicated in each panel.
Fig. 6.—Four different isodensity surfaces are shown at the end \((t = 19.4t_c)\) of the evolution of the \(n = 1.8\) polytrope. The value of the density at which each surface is drawn is indicated in each panel.
Figure 7a begins at $t = 6t_c$ because, before about $6t_c$ no coherent pattern speed is identifiable in the model. As has been explained in WT, at times $t < 6t_c$ the bar-mode pointing angle $\phi_2$ appears to vary randomly in time and is not correlated with radius. From $t = 6t_c$ to $11t_c$, the spiral pattern amplitude is growing exponentially and the pattern speed $\Omega_2$ is homing in on a well-defined value $\Omega_2|_{\text{linear}}$; this time period defines the linear growth regime discussed extensively by WT. The exponentially growing spiral soon reaches a nonlinear amplitude and, from $t = 11t_c$ to $20t_c$, the pattern speed $\Omega_2$ changes from its well-defined "linear" value to a less well defined value which we will identify with the model's final state as $\Omega_2|_{\text{ring}}$. As the pattern speed increases from $\Omega_2|_{\text{linear}}$ toward the value $\Omega_2|_{\text{ring}}$, the corotation radius $R_{\text{co}}$ moves well inside the initial model. It is during this time period that some mass is lost from the computational grid and the ring is formed. The behavior of $\Omega_2$ versus time for the $n = 0.8$ polytrope evolution looks qualitatively similar to Figure 7a, so it is not shown here. For both models, however, Table 2 summarizes pattern speeds and corotation radii in both the linear and ring formation phases of the evolution: column 2 lists the linear regime pattern speed; column 3 identifies the corotation radius in the linear regime (see also Figures 1 and 2); column 4 lists our estimated mean value of $\Omega_2$ from $11t_c < t < 19.42t_c$ for the $n = 1.8$ polytrope and from $7t_c < t < 12.3t_c$ for the $n = 0.8$ polytrope; and column 5 identifies the corotation radius in the final "ring" state. The quantity $\Omega_2|_{\text{ring}}$ is marked on Figure 7a and on Figure 8a (explained below); it is apparent that there is some freedom in choosing this value since $\Omega_2$ clearly oscillates during the latter part of the evolution. Choosing a larger value of $\Omega_2|_{\text{ring}}$ places corotation even closer to the center of the object.
than the distance \( R_{eo} \) listed in Table 2.

Figure 7b is a plot of \( \beta \) versus time for both polytropes. Notice how, late in the evolution, both \( \Omega_2 \) and the global parameter \( \beta \) for the \( n = 1.8 \) polytrope undergo a series of oscillations. (Gingold was the first to spot oscillations of \( \beta \) in an earlier, related study; see Durisen, Gingold, Tohline, and Boss 1986.) The oscillations in \( \beta \) damp with time, but the oscillations in \( \Omega_2 \) actually become more pronounced. The oscillations, which can be seen in the \( n = 0.8 \) model as well, are associated with radial pulsations of the central triaxial object (see below).

Figure 7c is a plot of \( J/J(t = 0) \) versus time for both polytropes, where \( J \) is the total angular momentum enclosed within the computational grid. Figure 7d is a plot of mass within the computational grid versus time for both polytropes. The downward arrows next to the letters 3a–3f on Figure 7c identify exactly the times at which the six isodensity surfaces in Figure 3 (frames a–f, respectively) have been drawn. The downward arrows next to the letters 4a–4f on Figure 7d reference, in a similar manner, the times at which the six isodensity surfaces in Figure 4 are drawn. As can be seen by comparing Figures 3 and 4 with Figures 7c and 7d, the main phase of mass and angular momentum loss via flow through the grid boundary occurs during the spiral elongation phase. After the spiral elongation phase is over, further redistribution of angular momentum via torques exerted by the central triaxial object continues to occur—along with some associated mass loss—but in a much less dramatic way.

Looking at hundreds of isodensity surfaces as they change in time reveals radial pulsations of the central triaxial object. We are working on a movie of this. (Equipment requirements and data manipulation make this far from a triv-
Fig. 7.—(a) Pattern speed versus time in central rotation periods for the $n = 1.8$ polytrope at the equatorial position $r/R_{eq} = 0.64$. (b) The parameter $\beta$ versus time for both models. (c) Normalized total angular momentum versus time for both models. (d) Mass within the computational grid versus time for both models. In frames b–d, the solid line is for the $n = 1.8$ model and the dashed line is for the $n = 0.8$ model.
Fig. 8.—(a) Pattern speed versus time for the $n = 1.8$ polytrope at the equatorial position $r/R_{eq} = 0.24$. (b) The natural logarithm of the normalized "bar-mode" amplitude $D_2$ versus time for the $n = 1.8$ polytrope at the equatorial position $r/R_{eq} = 0.240$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The existence of this pulsation can be confirmed by looking at the
time-variation of certain local diagnostic parameters. Figure 8a shows $\Omega_2/\omega_c$
versus time, and Figure 8b shows $\ln D_2$ versus time for the $n = 1.8$ polytrope
at $r/R_{eq} = 0.240$ which is within the central triaxial object. As explained in
WT, the parameter $D_2$ is a measure of the relative strength of the “bar-mode”
and is derived from a Fourier analysis of the azimuthal density distribution at a
particular radius in the equatorial plane of a model. Notice that both $\Omega_2$ and
$\ln D_2$ oscillates. During the nonlinear regime when the ring forms, the value of $\Omega_2$
increases when the value of $\ln D_2$ decreases. There also appears to be a gradual
secular increase in $\Omega_2$ after the pattern speed changes in the nonlinear regime.
These phenomena are consistent with the notion that during a single pulsation
cycle the central triaxial object essentially conserves its angular momentum (spin-
ing up when its moment of inertia decreases) but over many oscillation cycles
steadily transports some fraction of its angular momentum to the circumstellar
ring.

Figure 9 is a surface showing $\ln c_0/2$ versus both time and radius for the $n = 1.8$ model. The quantity $c_0/2$, derived from our Fourier analysis of the density
distribution, is a measure of the azimuthally averaged density at a particular
radial location in the equatorial plane of the model (see WT). Figure 9 shows that,
for $t \lesssim 10t_c$, the mass does not spread out radially. This early behavior reflects
the accuracy with which the initial hydrostatic structure was obtained. During
this time, though, the nonaxisymmetric modes are growing toward nonlinear
amplitudes. A comparison between this figure and Figure 3 shows that the
radial expansion of the model just after $t = 10t_c$ is associated with the nonlinear
development of the two-armed spiral. After the spiral elongation phase, ring formation along with a sudden increase in the maximum density of the central object can be clearly seen. The deep trough that develops between the ring and the central object is apparent at late times. Pulsations of the central object are also suggested at late times by the wave in the surface near \( r = 0 \).

At the end of the evolution of the \( n = 1.8 \) polytrope, 10% of the initial mass and 41% of the initial angular momentum have been lost from the computational grid. The central triaxial object contains 86% of the remaining mass, 52% of the remaining angular momentum, and has a major radius 0.3 times the original protostar size. The ring contains 14% of the remaining mass and 47% of the remaining angular momentum. The ring is centered at 0.6 times the original protostar size. The final configuration has a mean \( \beta \sim 0.20 \).

At the end of the evolution of the \( n = 0.8 \) polytrope, 6.7% of the initial mass and 30% of the initial angular momentum have been lost from the computational grid. The central triaxial object contains 99% of the remaining mass, 94% of the remaining angular momentum, and has a major radius 0.75 times the original protostar size. The ring contains 1% of the remaining mass and 4% of the remaining angular momentum. The ring is centered at 2.2 times the original protostar size. The final configuration has a mean \( \beta \sim 0.22 \).

Isodensity surfaces can be deceptive. Because it occupies a larger volume, the ring in Figure 4f looks more massive than the ring in Figure 3f, but the reverse is true in the extreme. Given equal total mass in the two final states, the ring in the \( n = 1.8 \) polytrope (Figure 3f) is 14 times more massive than the ring in the \( n = 0.8 \) polytrope (Figure 4f).
Fig. 9.—A measure of the azimuthally averaged mass density—$\ln c_0/2$—in the equatorial plane of the $n = 1.8$ model is shown versus time and radius.

$n=1.8, \beta = 0.310$
IV. DISCUSSION AND CONCLUSIONS

a) General Summary

As we reviewed briefly in §I, it has been known for a long time that rapidly rotating, self-gravitating gas clouds are susceptible to a variety of different dynamical instabilities. This work, along with a few other closely related recent works, has focused on analyzing the nonlinear outcome of the nonaxisymmetric, "bar-mode" instability that arises in an axisymmetric, compressible system when the ratio of rotational to gravitational energy $\beta$ is initially $\geq \beta_d$. The outcome of this instability seems now to be very well established. A coherent, nonaxisymmetric "bar" does not amplify and lead to dynamical fission of the system into a close binary pair; instead, a two-armed spiral pattern grows dynamically to nonlinear amplitude and an associated rapid redistribution of angular momentum leads to the shedding of high specific angular momentum material in the equatorial plane of the system. At the end of the period of dynamical mass-shedding, a rapidly rotating, centrally condensed triaxial object is formed and some fraction of the excreted material becomes organized as a fairly axisymmetric, circumstellar ring which lies in the equatorial plane not far outside the central object.

Qualitatively, this result does not seem to be sensitive to the exact value of $\beta$ in the initial configuration (although, certainly, it must be $> \beta_d$), and it does not seem to be sensitive to the degree of compressibility of the system's equation of state. In this sense, the result is very robust. Quantitatively, however, the outcome of a particular cloud evolution does depend on both $\beta_i$ and the polytropic index $n$. Table 3 summarizes, for example, the fractional mass loss and angular momentum loss that has been observed to occur in the two evolutions described...
in this paper, in the two evolutions discussed by Durisen, Gingold, Tohline and Boss (1986), and in the single evolution reported by Durisen and Gingold (1986). From this data, some trends are apparent: axisymmetric clouds with larger $\beta_i$ lose a larger fraction of their angular momentum and mass during the phase of equatorial mass shedding; and more compressible structures—polytropes with larger $n$—lose more matter and angular momentum. It is also clear from the present study that, for a given $\beta_i$ and initial equatorial radius $R_{eq}$, the final triaxial object will have a larger major radius when a stiffer equation of state is employed. The radius of the circumstellar ring will be correspondingly larger as well.

It does not appear that either the existence of or the early linear-amplitude development of a dynamically unstable, two-armed spiral mode can be physically attributed to the favorable location of the corotation radius or the Lindblad resonances in these initial models. The Lindblad resonances cannot be important because, as we pointed out in §II, neither an inner nor an outer Lindblad resonance appears within the initial axisymmetric model configuration. The insignificance of corotation is attested to by Figure 2: for the $n = 0.8$ model, corotation lies outside of the volume of the initial, axisymmetric model. During the phase of each evolution when the two-armed spiral pattern reaches nonlinear amplitude and then experiences significant radial expansion (see panels b–d of Figures 3 and 4), it is much more difficult to assess the role of these classical resonance points. During this rapid phase of the evolution, the spiral pattern elongates and material flows well beyond the initial corotation radius in both models. Material also flows beyond the radius where the outer Lindblad resonance would have been
identifiable in an axisymmetric, zero-mass Keplerian disk surrounding the initial models. It is during this phase of both evolutions that significant redistribution of angular momentum occurs. At the time when a significant amount of material flows radially to large distances in the models, however, both the gravitational potential field and the fluid velocity field exhibit strong nonaxisymmetric components. It is therefore difficult to identify the location of the classical resonances during this phase of each evolution. Furthermore, even if they could be identified, it is not immediately obvious that locating resonance points which are defined by a linear perturbation theory can be a useful step toward explaining, for example, the process of redistribution of angular momentum in a highly nonlinear system.

Curiously enough, once the primary dynamical phase of each evolution is over and the final “central triaxial object plus circumstellar ring” structure has been established, the corotation radius does appear to play an important role. The vivid radial gap that appears between the central object and the ring seems to be located very near, if not exactly at, the corotation radius of the final configuration. (Again, the exact location of corotation is difficult to specify because the amplitude of the central triaxial object is nonlinear and, hence, the flow fields it generates have significant nonlinear components. In addition, as a result of the global dynamical pulsations exhibited by the central object, the pattern speed and the fluid velocity field of the final configuration is time-varying, so the location of corotation is rather ill defined.) The importance of the corotation resonance to the final state of the system is exemplified not only by the cleanness of the gap but also by the presence of a well-defined radial shock that develops at the inner edge of the circumstellar ring.
One interesting aspect of our models that should be stressed is the pulsational characteristic of the final central triaxial figure. Interest in this characteristic does not stem from an expectation that the pulsations will have a significant impact on the long-term evolution of the system. For example, the continued redistribution of angular momentum and material that is observed to proceed on a secular time scale at the end of our evolutions is almost certainly due to the nonaxisymmetric—not the pulsational—nature of the central object as it exerts a gravitational torque on the orbiting circumstellar material. Instead, our interest in the pulsations stems from a general concern about the equilibrium structure of self-gravitating, triaxial objects. In the past, attempts to construct hydrostatic models of any equilibrium, triaxial structure having both a high $\beta$ value and a compressible equation of state have met with very limited success. Uniformly rotating models with a polytropic index $n = 0.8$ have been constructed with $\beta$ as high as 0.14 but, due to the excessive centrifugal force felt by material at the tip of the major axis of objects that are forced to rotate uniformly, it is impossible to construct $n = 0.8$ polytropic models with $\beta > 0.14$. For uniformly rotating, triaxial objects having higher degrees of compressibility, such as $n = 1.8$ polytropes for example, the upper limit on $\beta$ is much lower than 0.14. It has been clear for some time, then, that rapidly rotating, compressible triaxial objects must generally have nontrivial internal motions. Attempts to construct models of such objects by using standard hydrostatic modeling techniques (see, for example, the three-dimensional self-consistent-field code developed by Hachisu 1986) have been thwarted by a lack of understanding of how to represent complex internal motions in a physically realistic way. What our current model evolutions
have demonstrated is that compressible triaxial figures having fairly large values of $\beta$ can be constructed—a result that is interesting in itself—but the objects that we have inadvertently constructed using a hydrodynamic code invariably pulsate. We suggest, therefore, that a natural attribute of most rapidly rotating, gaseous, triaxial configurations may be perpetual pulsation and that, as a result, a continued search for hydrostatic analogues of such systems may prove to be a fruitless endeavor.

We should point out that, even though the evolutionary phase which exhibits a well-defined central triaxial figure is an interesting and important one in many respects, this phase of any gas cloud’s evolution is almost certainly destined to be a transient one. Through its gravitational interaction with the circumstellar ring/disk, the central object will continue to lose angular momentum. This angular momentum loss cannot continue indefinitely. It will most likely stop when the rotational energy of the central object becomes sufficiently small that a triaxial figure is no longer energetically favored and the central object relaxes into an axisymmetric state. Judging by the rate at which the variable $\beta$ is, on average, declining in Figure 7, we estimate that, at least in the case of the $n = 1.8$ polytrope, continued interaction between the triaxial figure and the circumstellar ring will drive the configuration back to an axisymmetric state in $\lesssim 100$ rotation periods.

b) Application to Protostars

As we examine what we have learned from our numerical simulations that will contribute specifically to our understanding of the evolution of rapidly rotating protostars, we will focus preferentially on the results obtained from the $n = 1.8$
model. This is reasonable since, during much of its evolution toward a stellar state, a protostar is expected to have a gaseous composition that exhibits an effective adiabatic exponent $\Gamma_{\text{eff}} \lesssim 5/3$ [i.e., $n = 1/(\Gamma_{\text{eff}} - 1) \gtrsim 3/2$].

Part of the original motivation for this study centered around the desire to determine whether or not the dynamical “bar-mode” instability that can arise in a rapidly rotating protostar will led to binary fission and, hence, provide a natural explanation for why a large fraction of stars reside in short-period binary systems (Abt 1978, 1987). The nonlinear, dynamical evolution that results from this instability does lead to fission, in the sense that there is a distinct segmentation of the original object into two pieces, but as we have just summarized, it is clear that the instability does not lead to binary fission.

Our evolutions draw special attention to the intimate relationship that can develop between a rapidly rotating protostar and a low-mass, circumstellar disk. We should expect, as a natural consequence of gravitational collapse from interstellar clouds, that most stars will form with some amount of high specific angular momentum debris left surrounding them in a disk. Besides the obvious indication that our own solar system had a disk of debris associated with it at an early age, direct and indirect observational evidence of gas/dust disks both around young stars and in protostellar environments supports this expectation (Welin 1978; Adams and Shu 1986; see Harvey 1985 for an excellent brief review.) One thing we have demonstrated is that, even if a substantial disk does not initially surround a rapidly rotating protostar, the protostar is likely to create its own disk during a stage of contraction toward the main sequence as it tries to deal with its inherent angular momentum problem. We have modeled
in a rigorous fashion the manner in which gravitational torques resulting from a
dynamically growing two-armed spiral structure can act as an effective tool for
angular momentum redistribution in a rapidly rotating protostar. In addition to
this we have demonstrated how, in a fully three-dimensional, nonlinear system,
redistribution of angular momentum can continue on a secular time scale via the
gravitational interaction between a protostar and its circumstellar ring (or disk)
if the protostar itself is triaxial. As we mentioned above, the role of a corotation
resonance is apparently quite important during this type of secular evolution.

It is tempting to draw a connection between the properties of our final $n = 1.8$
configuration and the general properties that would be required of a protostellar
object that can collimate its radiation field into a “bipolar flow.” In particular,
due to the strong resonance at corotation and the (imposed) adiabatic nature of
the gas, the inner edge of the circumstellar ring is well defined and geometrically
thick. The inner edge of the ring is substantially thicker, for example, than is
the central object along its minor axis. Although we do not currently follow
the radiation flow from the central protostar, it seems clear that the radial gap
between the central object and the ring will allow radiation to flow freely from
virtually the entire surface of the protostar but that the well-defined wall at
the inner edge of the ring will attenuate the flow over a sizeable solid angle in
the equatorial plane of the system. Therefore, a bipolar flow with a fairly wide
opening angle could conceivably develop. In practice, though, it is doubtful that
bipolar flows in protostellar gas clouds derive their collimation in any direct way
from an evolutionary state like the one we have demonstrated here because, as
we will discuss presently, the timescales involved are too disparate.
During the last phase of the evolution of the $n = 1.8$ polytropic model, as has been depicted in Figures 7 and 8, at a slow but steady rate high specific angular momentum material continues to flow off the computational grid, $\beta$ for the entire configuration continues to decrease, and the pattern speed of the central tumbling triaxial figure continues to increase. This secular evolution of the system is driven, as described above, by the gravitational interaction between the central object and the circumstellar ring. Because this phase of the evolution allows the central object to continue to lose some fraction of its angular momentum, it undoubtedly represents an important phase through which a rapidly rotating protostar can evolve and deal successfully with its angular momentum problem. Unfortunately, it is highly unlikely that this evolutionary phase will be directly observable in any protostar or protostellar gas cloud. The odds are against directly detecting this phase because, in practice, the fraction of a protostar's total evolutionary lifetime that should be spent in this phase will be very small. For example, we estimated above that this phase of the evolution would last for $\lesssim 100$ rotation periods. Adopting the specific protostellar cloud dimensions suggested in §II—that is, a $1 \, M_{\odot}$ cloud with an equatorial radius of 100 AU, an average hydrogen number density of $2 \times 10^{11} \, \text{cm}^{-3}$, and a central rotation period of 270 years—this translates into an evolutionary phase lasting $\lesssim 3 \times 10^4$ years. (If a rapidly rotating protostar encounters our modeled instability at an even later stage of its contraction toward the main sequence, the total time spent in this phase will be even smaller, scaling as $n_\text{H}^{1/2}$.) The low probability of direct observational detection should not, however, detract from the overall significance of this phase of a protostar's evolution.
c) Other Astrophysical Systems

It is important to appreciate that the model evolutions presented here have some relevance to dynamically evolving astrophysical systems other than protostars. Indeed, the calculations were performed in dimensionless “polytropic units” specifically so that their relevance to other systems could more readily be ascertained. Here we will briefly draw attention to possible connections between our models and barred spiral galaxies, and between our models and rapidly rotating neutron stars.

It has been thought for some time that the global features observed in the disks of barred spiral galaxies—for example, a prominent, two-armed spiral bending tightly off the ends of a central, coherent bar-like structure and, occasionally, the spiral wrapping back around on itself to form a pattern looking like the Greek letter theta—are probably driven in an approximately steady way by a tumbling barlike distortion that exists in the central regions of these galaxies. Our evolutions demonstrate, in a completely self-consistent way, how a tumbling barlike structure might form dynamically in a young galaxy before a substantial fraction of the gas in the galaxy has been converted into stars. Early on, the amplitude of the central barlike distortion would be expected to be substantially nonlinear and the corotation radius extremely well defined. As the galaxy system ages, however, the amplitude of the distortion would naturally subside to the levels observed in nearby barred spirals due both to the continued transfer of angular momentum out of the central regions and to the conversion of gas into stars. It is entirely possible that the barlike distortions still present today in these galaxies are the direct descendents of triaxial figures like the ones we have seen at the end.
of our simulations because, unlike the protostellar case discussed above, on the scale of an individual galaxy an evolutionary phase that lasts for $\sim 100$ rotation periods can represent a substantial fraction of a Hubble time. With this in mind, we raise the possibility that the barlike distortions that are believed to exist in the central regions of many spiral galaxies are not steady-state distortions. Instead, our results suggest that these distortions may have a substantial pulsation amplitude and that, during each pulsation, the central region oscillates between a roughly axisymmetric structure and one that is barlike. Clearly the spiral pattern driven by such an oscillating gravitational potential field will not be steady-state either, but will subside periodically. To our knowledge, this possibility heretofore has not been considered in studies that have attempted to model the structure of barred spiral galaxies.

If the core of a massive star rotates sufficiently rapidly that the neutron star formed from core collapse has a $\beta > \beta_d$, then, shortly after its birth, the neutron star is likely to evolve in a manner similar to that depicted here. Our $n = 0.8$ model evolution is particularly relevant since the corresponding equation of state for that model has an effective adiabatic exponent $\Gamma_{eff} = 2.3$, not unlike that expected of neutron stars. Based on this conjecture, as the supernova explosion associated with the formation of the neutron star is pushing most of the stellar envelope away from the core region, the neutron star itself will be able to "excrete" a circumstellar ring. We note, in this context, that Michel and Dessler (1981) have discussed a particular pulsar mechanism that relies on the existence of such a ring or disk of debris. Furthermore, the evolution would lead directly to the formation of a triaxial neutron star whose geometric figure is not stationary.
in inertial space. The triaxial figure would exhibit both rotation about its shortest axis and a *global pulsation*. Although this triaxial state, having a rotation period $< 1$ millisecond, would not be long-lived, its existence would presumably be discernable from the spectrum of any emitted gravity wave radiation. To ascertain what the spectral signature of pulsation on top of figure rotation would be in a very young neutron star, a relativistic treatment of the problem—at least in the post-Newtonian approximation—will have to be done. Since direct observational evidence of a pulsating, triaxial figure in a protostellar gas cloud will be practically impossible to obtain, though, detecting the signature of a pulsating, triaxial neutron star during a supernova explosion would be quite gratifying since it would at least lend support to the claimed generality of our models.

d) Future Work

There are a number of additional experiments that should be performed in a continued study of rotationally driven gravitational instabilities in order to further substantiate our findings. Some involve making further improvements in our numerical tools; others involve examining more completely the available physical parameter space.

In terms of improving our tools, we clearly need to remove the restriction of "pi-symmetry" from our currently employed grid in order to allow the growth of odd-numbered azimuthal modes. Modes with two-fold symmetry, such as the two-armed spiral in our current simulations, are expected to be the fastest growing ones in objects that are only marginally unstable—i.e., in models for which $\beta$ is only slightly larger than $\beta_d$—but this expectation needs to be verified. In models with $\beta$ sufficiently above $\beta_d$, including the effects of odd-numbered az-
Imuthal modes may be essential, as has been demonstrated in one case by Durisen and Gingold (1986). We also need to improve the general differencing scheme that has been employed to approximate the differential equations governing fluid flow. As we pointed out in §II, our current scheme is accurate only to first order in the spatial differences and, as a result, the scheme has inherent in it a measure of numerical viscosity that tends to fight the growth of nonaxisymmetric structures, particularly those structures having short wavelengths (see WT for a detailed discussion). By developing a scheme that is accurate to second order, for example, the code will be much less diffusive, allowing us to study smaller-scale azimuthal structures (high-order azimuthal modes) and to examine instabilities in models with $\beta_i$ very close to $\beta_d$. A numerical code which implements the second-order-accurate, van Leer monotonic interpolation scheme in three dimensions has been written and is currently being tested expressly for this purpose (Williams and Tohline 1987c). Although we do not expect the qualitative results of our model evolutions to change substantially when a second-order-accurate differencing scheme is implemented, we do expect some quantitative differences to arise. For example, the amount of material that truly escapes from the system versus the amount of material that finds its way into the final ring structure may very well depend on the influence of high-order azimuthal modes.

In probing additional regions of the physical parameter space, we see a need to examine the stability of models having a variety of different radial angular momentum distributions, such as would be represented by a power law other than $q = 3/2$ in the $j(r)$ relation given in §II, and to examine the nonlinear, adiabatic evolution of models which are initially not homentropic. A few studies
of this nature are already under way. Eventually, we hope to be able to model radiation flow from our dynamically evolving models so that a strictly barotropic equation of state will not have to be forced onto the system and so that a more direct connection between our work and observational studies can be made.

Numerous discussions with R.H. Durisen throughout our efforts are happily acknowledged. The bulk of the computing was carried out on LSU's System Network Computing Center's (SNCC) FPS-264s, front ended by an IBM-3084QX4. The assistance of the SNCC staff is greatly appreciated. The graphical analysis, which is so important in making sense out of multidimensional hydrodynamic simulations, was done in part using the NCAR plotting package on the IBM-3084. Monika Lee's assistance in implementing the graphics package is appreciated. Robert Flammang and Jeff Anderson also have been very helpful in managing the data and in running graphics programs. This work has been supported, in part, by the National Science Foundation through grants AST 85-01842 and AST 87-01503.
### TABLE 1

**INITIAL POLYTRPOE STRUCTURE**

<table>
<thead>
<tr>
<th>$n$</th>
<th>$R_{eq}$</th>
<th>$R_{eq}/R_p$</th>
<th>$\rho_c$</th>
<th>$\rho_c/\bar{\rho}$</th>
<th>$\omega_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.8...</td>
<td>39.3</td>
<td>5.62</td>
<td>$1.92 \times 10^{-4}$</td>
<td>8.74</td>
<td>$1.47 \times 10^{-2}$</td>
</tr>
<tr>
<td>0.8...</td>
<td>2.55</td>
<td>4.44</td>
<td>$1.69 \times 10^{-1}$</td>
<td>2.65</td>
<td>$4.54 \times 10^{-1}$</td>
</tr>
</tbody>
</table>

**Note.**—Polytropic units.

### TABLE 2

**HYDRODYNAMIC RESULTS**

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\Omega_c/\omega_c$</th>
<th>$R_{co}/R_{eq}$</th>
<th>$\Omega_c/\omega_c$</th>
<th>$R_{co}/R_{eq}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.8...</td>
<td>0.348</td>
<td>0.88</td>
<td>0.8</td>
<td>0.1</td>
</tr>
<tr>
<td>0.8...</td>
<td>0.405</td>
<td>1.2</td>
<td>0.6</td>
<td>0.7</td>
</tr>
</tbody>
</table>

### TABLE 3

**RESULTS OF FISSION SIMULATIONS**

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\beta_i$</th>
<th>Mass Lost</th>
<th>Ang. Mom. Lost</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.33</td>
<td>10%</td>
<td>30%</td>
<td>Durisen and Gingold (1986)</td>
</tr>
<tr>
<td>1.5</td>
<td>0.33</td>
<td>18%</td>
<td>51%</td>
<td>Durisen et al. (1986)</td>
</tr>
<tr>
<td>1.5</td>
<td>0.38</td>
<td>36%</td>
<td>77%</td>
<td>Durisen et al. (1986)</td>
</tr>
<tr>
<td>0.8</td>
<td>0.31</td>
<td>8%</td>
<td>36%</td>
<td>This paper</td>
</tr>
<tr>
<td>1.8</td>
<td>0.31</td>
<td>23%</td>
<td>70%</td>
<td>This paper</td>
</tr>
</tbody>
</table>
APPENDIX C
SECOND-ORDER 3-D EXPLICIT EUCLERIAN HYDRODYNAMIC
COMPUTER CODE LIST IN FORTRAN

C.1 INTRODUCTION

The following is a list of the FORTRAN source code for my second-order 3-D
explicit Eulerian hydrocode. The FORTRAN source code is divided into four
subroutine collections: MAININ, MAINCN, HYD3D20, and POTS.

The subroutine collection called BLKTRI is not listed, even though it is
needed by POTS in the Poisson solver. BLKTRI subroutines are FORTRAN
subprograms for solving the system of linear equations which usually results
from discretization of separable elliptic partial differential equation:

\[
AN(K) \times X(J, K - 1) + AM(J) \times X(J - 1, J) + \\
[BN(K) + BM(J)] \times X(J, K) + \\
CN(K) \times X(J, K + 1) + CM(J) \times X(J + 1, K) = Y(J, K),
\]

for \( J = 1, 2, \ldots, J_{\text{max}} \) and \( K = 1, 2, \ldots, K_{\text{max}} \), where \( J \pm 1 \) is evaluated modulo
\( J_{\text{max}} \) and \( K \pm 1 \) modulo \( K_{\text{max}} \); i.e.,

\[
X(J, 0) = X(J, K_{\text{max}}); \quad X(J, K_{\text{max}} + 1) = X(J, K);
\]

\[
X(0, K) = X(J_{\text{max}}, K); \quad X(J_{\text{max}} + 1, J) = X(1, K),
\]

BLKTRI is a member of FISHPACK which is available from NETLIB @ ANL-MCS.arpa over the network. BLKTRI is documented in Swarztrauber and Sweet
(1975), and the solution of the linear equation (A.1) that results from discretiza-
tion of separable elliptic equation is explained in Swarztrauber (1974). FISH-
PACK was produced at NCAR.
The subroutine collection called FFTPACK, which are FORTRAN subprograms for performing fast Fourier transforms, is not listed, even though it is needed by POTS in the Poisson solver. FFTPACK is also available from NETLIB @ ANL-MCS.arpa over the network. FFTPACK was written by Swarztrauber at NCAR.

Most supercomputer manufacturers now provide FFT routines written in machine code and optimized for their computer’s architecture. Once I substituted FPS routines on the FPS M64/60 (FPS used to call this machine the FPS-264) for FFTPACK routines in the middle of a model simulation. The model was a rapidly rotating, centrally condensed polytrope, which was initially in axisymmetric equilibrium and had $\beta = 0.313$, $n = 1.8$, and $q = 3/2$. The FPS routines ran a little faster, but the gravitational energy decreased somewhat and the model started to form a binary. When I reran the model using FFTPACK from before the point of the FPS substitution, no binary formed. It is not surprising that a sudden decrease in the gravitational potential of a rapidly rotating polytrope would cause it to immediately undergo binary fission. Unfortunately for those of us who would like to understand binary formation, nature does not do this. I do not know whether the FPS FFT routines are in error or if they are just of different precision. FFTPACK has been very well tested, for more than a decade, and the time increase was not very significant. I conclude that one should never change FFT routines in the middle of a calculation. This is another sign of how careful one has to be while doing a hydrodynamic calculation, if you are to believe the results. I believe my results, but I try to be very careful. Any change, no matter how trivial and equivalent it may appear to be to the previous way of
doing things, must be carefully checked.
THIS PROGRAM IS DESIGNED TO HANDLE ROTATING POLYTOPES WITH AN
EULARIAN GRID WHICH ROTATES BUT DOES NOT MOVE IN R AND Z.
RIGHT NOW UNIFORM ROTATION OF THE GRID IS DISABLED.
SYMMETRY ABOUT THE EQUATORIAL PLANE IS ASSUMED BUT SYMMET
ABOUT THE ROTATION AXIS IS EITHER PI SYMMETRY OR 2PI, NO SYMMET
ABOUT THE ROTATION AXIS.
IZUMI POLYTROPES ARE THE INITIAL MODEL. WILLIAMS FORMATED INST
OF TOHLINE FORMATED.
SECOND ORDER MODELS ARE RUN.
JN AND OMEGA STILL NEED TO BE REMOVED.
AND WRITTEN OUT AS S,T,A,RHO,ROF3N,ZOF3N,DELT,TIME,MLOSTR,MLOST
JLOSTR,JLOSTZ.

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)

COMMON /STEP/ ICNT
COMMON /BL0K6/ DTTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAV
COMMON /BL0K7/ RCL0UD,CONSTP,DELT,BDYTEM,DEN,TIME,CORMAS
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX1),
1 ROF3N,ZOF3N,A1NEWR,A1NEWZ
REAL JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /INSIDE/ TMASS,ENEW,ELOST,EDIF,PHICHK,KLOCAT
COMMON /TIMST/ INDX,ISOADI,ALLOW,ITSTEP
CHARACTER*6 BOTBDY,TOPBDY,SIDBDY
INTEGER MINUS1,ZERO,ONE, TWO,THREE, FOUR,FIVE,EIGHT,F15,FIFTY,
1 LENG1,LENG2
LENG2=8*(4*JMAX2*KMAX2*LMAX+10)=1253456
OPEN( UNIT=7, FILE='DIAG07,LENGTH=132', STATUS='NEW' )
OPEN( UNIT=9, FILE='MOD09,LENGTH=80', STATUS='OLD' )
OPEN( UNIT=12, FILE='MOD12,LENGTH=120', STATUS='NEW' )
OPEN( UNIT=13, FILE='MOD13,LENGTH=1253456', FORM='UNFORMATTED',
1 STATUS='NEW' )
OPEN( UNIT=25, FILE='COEF25,LENGTH=120', STATUS='NEW' )
MINUS1=-1
ZERO=0
ONE=1
TWO=2
THREE=3
FOUR=4
FIVE=5
EIGHT=8
F15=15
FIFTY=50
ISDIAG=5
INDX=0
CALL SETUP(ITSTRT, ITSTOP, IDIAG, ISOADI, ISTOR, ITYPE, ISYM, MAXTR,
           BOTBDY, TOPBDY, SBDY)
CALL GEM(ISYM, BOTBDY, TOPBDY, SBDY)
C WHAT IS THE PURPOSE OF THE BELOW IF STATEMENT.
   IF(RHO(JMAX2, KMAX1, 1).EQ.0.0) RHO(JMAX2, KMAX1, 1) = 1.0
   CALL RITE(THREE, MINUS1, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
   ITSTOP=ITSTOP-1
   DO 90 ITSTEP=ITSTRT, ITSTOP
      ICNT=ITSTEP
      INDX=ITSTEP-ITSTRT
      IHEAD=0
      IF(MOD(INDX, IDIAG).EQ.0) IHEAD=-1
      IPRINT=0
      IF(MOD(INDX, IDIAG).EQ.0) IPRINT=1
      CALL DELTA
      CALL SOST(ISYM, BOTBDY, TOPBDY, SBDY)
      CALL SOA(ISYM, BOTBDY, TOPBDY, SBDY)
      CALL VIS
      CALL VEL
      CALL HYDROS(ISYM, BOTBDY, TOPBDY, SBDY)
      C** CALL VEL
      IF(INDX.EQ.0) GO TO 40
      IF(MOD(INDX, ISTOR).EQ.0) CALL RITE(TWO, MINUS1, ITSTEP, ITS
1
   40 CONTINUE
   CALL EOC(ISYM, BOTBDY, TOPBDY, SBDY)
   CALL VEL
      IF(MOD(INDX, IDIAG).NE.0) GO TO 47
      CALL RITE(THREE, MINUS1, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
      CALL RITE(ONE, MINUS1, TWO, JMAX2, ONE, TWO, TWO, ONE, ONE, LMAX
1
   44 CONTINUE
   47 CONTINUE
   IF(MOD(INDX, ISDIAG).EQ.0) CALL RITE(FOUR, ONE, ONE,
1
   CALL STATE(THREE, ZERO, ZERO, ZERO)
      REDGE=R(JMAX1)
      CALL BDYGEN(MAXTRM, ISYM, REDGE)
      CALL POT3(EIGHT, IPRINT, ISYM)
      IF(MOD(INDX, FIFTY).EQ.0) CALL RITE(FIVE, ISYM, ICNT,
1
   CALL RITE(ZERO, IHEAD, ITSTEP, INDX, ISYM, ONE, ONE, ONE, ONE, ONE, IS
90 CONTINUE
   ITSTEP=ITSTOP+1
   CALL RITE(FIVE, ISYM, ITSTEP, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
   CALL RITE(THREE, MINUS1, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
   CALL RITE(TWO, MINUS1, ITSTEP, ITSTRT, ITSTOP, ISTOR, ONE, ONE,
1
   90 CONTINUE
      CLOSE(7)
      CLOSE(9)
      CLOSE(12)
      CLOSE(13)
CLOSE(25)
STOP
END

SUBROUTINE RITE(IWHAT, IHEAD, JST, JSP, JSK, KST, KSP, KSK, LST, LSP,
  C  IWHAT = 1 PRINTS ALL VARIABLES OUT IN LPE12.4 FORMAT.
  C  = 0 BRIEF DIAGNOSTICS (2 LINES) ONLY AND COEF FILE OF
  C  = 2 STORES THIS MODEL.
  C  ABS(IHEAD) = 1 PRINTS HEADING ONCE PER CALL TO SUBROUTINE.

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
  1  KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
  2  LMAX=64)

COMMON /BLOK6/  DTHETA, COSIGN(LMAX), SIGN(LMAX), PI, GRAV
COMMON /BLOK7/  RCL OUD, CONSTP, DE L T, B D Y TEM, DEN, TIME, COR M AS
COMMON /NORMAL/  CIRP
COMMON /GRID/  R(JMAX2), Z(KMAX2), RHF(JMAX2), ZHF(KMAX1),
  1  ROF3N, ZOF3N, ALNEWR, ALNEWZ
REAL
COMMON /EOM/  S(JMAX2, KMAX2, LMAX), T(JMAX2, KMAX2, LMAX),
  1  A(JMAX2, KMAX2, LMAX), U(JMAX2, KMAX2, LMAX),
  2  W(JMAX2, KMAX2, LMAX), JN(JMAX2, KMAX2, LMAX),
  3  OMEGA(JMAX2, KMAX2, LMAX), VT(JMAX2, KMAX2, LMAX)
COMMON /STATES/  P(JMAX2, KMAX2, LMAX)
COMMON /TEMPCV/  EPS(JMAX2, KMAX2, LMAX)
C*
COMMON /POIS/  PHI(JMAX2, KMAX2, LMAX), RHO(JMAX2, KMAX2, LMAX)
COMMON /USE2/  RD(JMAX2), RDOLD(JMAX2), RDB(JMAX2),
  1  RINV(JMAX2), R3(JMAX2), RDM(JMAX2),
  2  RDP(JMAX2), GRADR(JMAX2), ZD(KMAX2),
  3  ZDOLD(KMAX2), ZDB(KMAX2), ZDM(KMAX2),
  4  ZDP(KMAX2), GRADZ(KMAX2)
COMMON /INSIDE/  TMAS S, ENEW, ELOST, EDIF, PHICHK, KLOC AT
COMMON /INRITE/  EGOLD, EKOLD, PDVOLD
COMMON /COEFS/  COE F(JMAX, LMAX, 2)
REAL
COMMON /ANGMOM/  JLO STR, JLOSTZ, MLO STR, MLOSTZ
COMMON /SUMRY/  EM(JMAX1), ENRTIA(JMAX1), EJ(JMAX1),
  1  ERO T(JMAX1), EG(JMAX1), BETA(JMAX1)
COMMON /ROTFRM/  OMGFRM
REAL JN INT E R
DIMENSION  RO(JMAX2), ZO(KMAX2), RHFO(JMAX2), ZHFO(KMAX1)
EQUIVALENCE (RO(1), R(1))
EQUIVALENCE (ZO(1), Z(1))
EQUIVALENCE (RHFO(1), RHF(1))
EQUIVALENCE (ZHFO(1), ZHF(1))
DIMENSION  WORK(JMAX1, 6)
EQUIVALENCE  (WORK(1, 1), EM(1))
**OMGFRM** set here because on a continued model this card is not required.

**OMGFRM** = 0.0

100 FORMAT(1HL)
101 FORMAT(//,' J K L ',5X,'JN',9X,'OMEGA',8X,'RHO',9X,'PHI',
      1X,5X,'JN-INERTIAL',1X,'OM-INERTIAL',7X,'U',11X,'W'/)
102 FORMAT(1I4,1P8E12.4)
103 FORMAT(//)
104 FORMAT( ' TSTEP',3X,'TIME',8X,'DELT',6X,'ETOT/JT',4X,'EGRAV/
      1X,2X,'EKF/RKF',3X,'ALPHA/CD',3X,'EDIF/DMAX',3X,'ELOST/
      2X,6X,'TMASS',6X,'PHICHK',4X,'K'/)
105 FORMAT(I5,1P10E12.4,I4)
106 FORMAT(5X,1PE12.4,' CIRPS',6X,1P5E12.4,I4,2I3,2X,1P2E12.4,/)  
107 FORMAT(10X,'MLOST =',1P2E12.4,4I1X,'S MODEL FROM TIME STEP NUMBER',I5,' HAS
      2 S BEEN STORED ON DISK. IT IS STORED AS S',//,'S MODEL
      3',I2,' ON TAPE UNIT',I3,'...S',//,'S',4I1X,'S',//,
      4' SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
WRITE(7,102) J,K,L,JN(J,K,L),OMEGA(J,K,L),RHO(J,K,L),PHI(J,K,L)
JNINER,OMINER,U(J,K,L),W(J,K,L)

40 CONTINUE
GO TO 999

50 CONTINUE
ISYMA=IABS(JSK)
IF(IWHAT.NE.0) GO TO 95

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C WHAT = 0,
C
C PRINT OUT BRIEF SUMMARY OF GLOBAL VARIABLES,
C AND (AS OF 3/20/85 AT LSU) WRITE FFT COEFS TO DISK.
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

IF(IHEAD.LT.0) WRITE(7,100)
IF(IHEAD.GT.0) WRITE(7,103)
IF(IABS(IHEAD).EQ.1) WRITE(7,104)

C* WIRD NOW PUT IN SUBROUTINE DELTA

TCIRP=TIME/CIRP
DO 55 J=2,JMAX1

JP=J+1
RD(J)=R(JP)**2-R(J)**2

55 CONTINUE

C* FIND EGRAV,TOTJ,EROT.

AREA=0.5*DTHETA
EEG=0.0
TOTJ=0.0
ER=0.0

DO 60 L=1,LMAX

DO 60 K=2,KMAX

AREA2=AREA*ZD(K)

DO 60 J=2,JMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

DO 60 L=1,LMAX

DO 60 K=2,KMAX

60 CONTINUE
C... FIND RZKIN
RZK=0.0
DO 70 L=1,LMAX
DO 70 K=2,KMAX1
AREA1=AREA*ZD(K)
DO 70 J=3,JMAX1
VOL=RD(J)*AREA1
RZK=RZK + 0.5*VOL*(U(J,K,L)*S(J,K,L) + W(J,K,L)*T(J,K,L))
70 CONTINUE
AREA1=PI*RHFO(2)**2
DO 75 K=2,KMAX1
VOL=AREA1*ZD(K)
RZK=RZK + 0.5*VOL*W(2,K,1)*T(2,K,1)
75 CONTINUE
C
C... MULTIPLY INTEGRAL PROPERTIES BY FACTOR, DEPENDING ON
C SYMMETRY BEING USED.
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)FACT0R=1.0
IF(ISYMA.EQ.2.OR.ISYMA.EQ.9)FACT0R=2.0
IF(ISYMA.EQ.3)FACT0R=4.0
EGRAV=FACT0R*EEG
TOTJN=FACT0R*TOTJ
EEROT=FACT0R*ER
RZKIN=FACT0R*RZK
EKIN=EEROT+RZKIN
JRHO=2
KRHO=2
LRHO=1
RHOMAX=RHO(2,2,1)
DO 80 L=1,LMAX
DO 80 K=2,KMAX1
DO 80 J=2,JMAX1
IF(RHO(J,K,L).LE.RHOMAX)GO TO 80
RHOMAX=RHO(J,K,L)
JRHO=J
KRHO=K
LRHO=L
80 CONTINUE
DMAX=RHOMAX
JD=JRHO
KD=KRHO
LD=LRHO
CD=RHO(2,2,1)
PDV=0.0
ETOT=EGRAV+EKIN+ENEW+ELOST-PDV
ECHECK=0.0
IF(JSP.GE.0) GO TO 87
ECHECK=(EKIN-EKOLD + EDIF)/(EGOLD-EGRAV + PDV-PDVOLD)-1.0
87 EGOLD=EGRAV
PDVOLD=PDV
EKOLD=EKIN
C
C... INSERTS BY JOEL AT LSU 5/16/83 TO WRITE ALPHA, BETA, ETC.
EGG = ABS(EGRAV)
ETOT = ETOT/EGG
EKIN = EKIN/EGG
ALPHA = ENEW/EGG
EDD = EDIF/EGG
ELL = ELOST/EGG
BBETA = EEROT/EGG

C... STOP INSERT.
C WRITE(7,105) JST, TIME, DELT, ETOT, EGRAV, EKIN, ALPHA, EDD, ELL, TMA
1   PHICH, KLOCAT
WRITE(7,106) TCIRP, TOTJN, BBETA, RZKIN, CD, DMAX, JD, KD, LD, ECHECK

C... INSERT BY JOEL AT LSU 4/17/85 TO WRITE ALL COEF TO DISK FILE
C
IF(MOD(JST,LSK).NE.0) GO TO 999
IF(LMAX.EQ.1) GO TO 1294
LHAF = 6
WRITE(25,2001) JST, TIME, DELT, ETOT, EGRAV, EKIN, ALPHA, EDD, EMASS
WRITE(25,2002) TCIRP, TOTJN, BBETA, RZKIN, CD, DMAX
WRITE(25,2003) JD, KD, LD, JMAX, KMAX, LMAX, LHAF
WRITE(25,2004)((COEF(JE,MM,III), MM=1,LHAF),111=1,2),
   JE=2,JMAX)
1294 CONTINUE

C... STOP INSERT.
C
WRITE(7,107) JLOSTR, JLOSTZ, MLOSTR, MLOSTZ
IF(LMAX.EQ.1) GO TO 999
LHAF = 8
WRITE(7,121) (JE,((COEF(JE,MM,III), MM=1,LHAF),III=1,2),
   JE=2,JMAX,2)
GO TO 999
95 CONTINUE
NTAPES = 12
IF(IWHAT.NE.2) GO TO 300

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C IWHAT = 2,
C STORE ENTIRE MODEL ON DISK (UNIT NTAPES = 12).
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
ITSTEP = JST
ISTRT = JSP
ISTOP = JSK
ISTOR = KST
INDX = ITSTEP - ISTRT
NUM = INDX / ISTOP

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
IF(MOD(INDX,ISTOR).NE.0) NUM=NUM+1
IF(ITSTEP.GT.ISTOP) GO TO 250
IF(NUM.GE.3) GO TO 999

250 CONTINUE
IF(NUM.GT.3) NUM=3
JN(1,KMAX2,1) = JLOSTR
JN(2,KMAX2,1) = JLOSTZ
JN(3,KMAX2,1) = MLOSTR
JN(4,KMAX2,1) = MLOSTZ

WRITE(NTAPES,2222)U,W,JN,RHO, EPS,ROF3N,ZOF3N,ALNEWR,ALNEWZ,
1DELT, TIME, ELOST
WRITE(13)U , W , JN,RHO,R0F3N,Z0F3N,ALNEWR,ALNEWZ,
1DELT, TIME, ELOST
WRITE(13) S , T , A, RHO, ROF3N, ZOF3N, ALNEWR, ALNEWZ,
1DELT, TIME, ELOST
WRITE(7,110) ITSTEP, NUM, NTAPES

300 CONTINUE
IF(IWHAT.NE.3) GO TO 400
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C WHAT = 3 ,
C
C PRINT OUT CURRENT GRID STRUCTURE.
C
WRITE(7,115) ROF3N,ALNEWR,CORMAS
WRITE(7,116) R,RHF
WRITE(7,117) ZOF3N,ALNEWZ
WRITE(7,116) Z,ZHF

400 CONTINUE
IF(IWHAT.NE.4) GO TO 450
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C WHAT = 4 ,
C
C PRINT OUT RADIAL BEHAVIOR OF RHO AND VELOCITIES AT
C
A) K = 2 ,
B) J = JRHO,

C BOTH BEING DONE AT L = IHEAD (OR L = LRHO IF IHEAD = 0).
C
WRITE(7,118)
LL=IHEAD
RHOMAX=0.0
DO 405 L=1,I MAX
DO 405 K=2,K MAX
DO 405 J=2,J MAX
IF(RHO(J,K,L).LE.RHOMAX)GO TO 405
RHOMAX=RHO(J,K,L)
JRHO=J
KRHO=K
LRHO=L
405 CONTINUE
IF(LL.EQ.0) LL=LRHO
IF(LL.EQ.I OR LL.GT.LMAX) LL=1
WRITE(7,125)
DO 411 KK=2,KMAX1
WRITE(7,119) KK,JRHO,R(KK),RHO(KK,2,LL),PHI(KK,2,LL),OMEGA(KK 1
U(KK,2,LL)),Z(KK),RHO(JRHO,KK,LL),OMEGA(JRHO,KK,LL),
2 ,KK,LL)
411 CONTINUE
KSTRTJ=KMAX1+1
IF(JMAX1.LE.KMAX1) GO TO 414
DO 412 KK=KSTRTJ,JMAX1
WRITE(7,119) KK,JRHO,R(KK),RHO(KK,2,LL),PHI(KK,2,LL),
1 OMEGA(KK,2,LL),U(KK,2,LL)
412 CONTINUE
414 CONTINUE
450 CONTINUE
C
C
IF(IWHAT.NE.5) GO TO 500
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
IWHAT = 5,
C
PRINT OUT 'SUMMARY' MODEL OF ENERGIES, ETC., ON CYLINDRICAL SHELLS.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
ISYM = IHEAD
AREA=0.5*DTHETA
ISYMA=IABS(ISYM)
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8) FACTOR=1.0
IF(ISYMA.EQ.2.OR.ISYMA.EQ.9) FACTOR=2.0
IF(ISYMA.EQ.3) FACTOR=4.0
C
DO 455 J=2,JMAX1
JP=J+1
RD(J)=R(JP)**2-R(J)**2
455 CONTINUE
DO 457 K=2,KMAX1
KP=K+1
ZD(K)=Z(KP)-Z(K)
457 CONTINUE
C... FIND EGRAV, TOIJ, EROT.
DO 459 J=1,JMAX1
DO 459 I=1,6
WORK(J,I)=0.0
459 CONTINUE
DO 462 J=2,JMAX
AREA1=AREA*RD(J)
RPI2=RHP(J)**2
DO 460 L=1,LMAX

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 460 K=2,KMAX
VOL=ZD(K)*AREA1
EMAS=RHO(J,K,L)*VOL
ERTIA=EMAS*RPI2
EM(J)=EM(J) + EMAS
EJ(J)=EJ(J) + A(J,K,L)*VOL
ENRTIA(J)=ENRTIA(J) + ERTIA
EROT(J)=EROT(J) + 0.5*ERTIA*OMEGA(J,K,L)**2
EG(J)=EG(J) + 0.5*EMAS*PHI(J,K,L)
460 CONTINUE

C.....SINCE THIS SUMMARY GIVES MASS, ANGULAR MOMENTUM, ETC. 'INTER
C TO A GIVEN RADIUS, MUST MULTIPLY CURRENT (K,L) SHELL BY SYMM
C 'FACTOR' AND THEN ADD IN ALL PREVIOUS RADIAL SHELLS.
C
DO 461 I=1,5
461 WORK(J,I)=FACTOR*WORK(J,I)+WORK(J-1,I)
462 CONTINUE

WRITE(7,140)JST, TIME
WRITE(7,141)
WRITE(7,142)(J,RHF(J),(WORK(J,I),I=1,6),J=2,JMAX)

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64, MWSAVE = 4*LMAX + 15)

COMMON /BLOK6/ DTHETA, COSIGN(LMAX), SIGN(LMAX), PI, GRAV
COMMON /BLOK7/ RCLD, CNSTP, DELT, DBYTEM, DEN, TIME, CORMAS
COMMON /NORMAL/ CIRP
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX1), ZH(HF(KMAX1)),
1 ROF3N, ZOF3N, A1NEWR, A1NEWZ
COMMON /OLD/ RO(JMAX2), ZO(KMAX2), RHFO(JMAX1), ZHFO(KMAX1)
DIMENSION RO(JMAX2), ZO(KMAX2), RHFO(JMAX2), ZHFO(KMAX1)
EQUIVALENCE (RO(1), R(1))
EQUIVALENCE (ZO(1), Z(1))
EQUIVALENCE (RHFO(1), RHF(1))
EQUIVALENCE (ZHFO(1), ZHF(1))

DIMENSION EPS(JMAX2, KMAX2, LMAX)

REAL
COMMON /EOM/ S(JMAX2, KMAX2, LMAX), T(JMAX2, KMAX2, LMAX),
1 A(JMAX2, KMAX2, LMAX), U(JMAX2, KMAX2, LMAX),
2 W(JMAX2, KMAX2, LMAX), JN(JMAX2, KMAX2, LMAX),
3 OMEGA(JMAX2, KMAX2, LMAX), VT(JMAX2, KMAX2, LMAX)

COMMON /STATES/ P(JMAX2, KMAX2, LMAX),

COMMON /USE1/ EPS(JMAX2, KMAX2, LMAX)

COMMON /POIS/ PHI(JMAX2, KMAX2, LMAX), RHO(JMAX2, KMAX2, LMAX)

COMMON /USE2/ DIVS(JMAX2, KMAX2), DIVT(JMAX2, KMAX2),
1 DIVA(JMAX2, KMAX2), SAVE(JMAX2, KMAX2),
2 SAVEJ(JMAX2, KMAX2), SAVEA(JMAX2, KMAX2)

COMMON /INSIDE/ TMASS, ENEW, ELOST, EDIF, PHICHK, KLOCAT

REAL

COMMON /ANGMOM/ JLOSTZ, MLOSTZ, MLOSTZ

COMMON /Sbeta/ WSAVE(MWSAVE)

CHARACTER*6 BOTBDY, TOPBDY, SIDBDY

DATA CURLYR, GAMMA, XMU/83.14, 1.666667E0, 2.0/

100 FORMAT(1P5E15.8)
101 FORMAT(1H, 12X, 'CLOUD RADIUS = ', 1PE11.4, ' CM', //, 12X, 'CLOUD D
1 'GM CM-3', //, 5X, 'BOUNDARY TEMPERATURE = ', 1PE11.4
2 'DEGREES K', //, 10X, 'CONST. PRESSURE = ', 1PE11.4, ' DYNE/CM2',
3 'TIME STEP = ', 1PE11.4, ' YEARS', //, 13X, 'STARTING AGE = ', 1PE11.4
4 'YEARS', //, 9X, 'INITIAL ROTATION = ', 1PE11.4, ' SEC-1', ///)
102 FORMAT(12I5)
103 FORMAT('THIS WILL BE AN ISOTHERMAL COLLAPSE STARTING AT TIM
1 NUMBER', I4, ' AND', //, ' GOING THROUGH TIMESTEP NUMBER', I4, '.
2DIAGNOSTICS EVERY', I3, ' STEPS', //, '')
104 FORMAT('THIS WILL BE AN ADIABATIC COLLAPSE STARTING AT TIM
1 NUMBER', I4, ' AND', //, ' GOING THROUGH TIMESTEP NUMBER', I4, '.
2DIAGNOSTICS EVERY', I3, ' STEPS', //, '')
105 FORMAT(' READING IN MODEL NUMBER', I3, ' FROM TAPE UNIT', I3, '.

106 FORMAT(//, 10X, 'ROTATING POLYTROPE... N = ', 0PF5.2, //, 27X, '1 + 1
1 1PE10.3, //, 33X, 'K = ', 1PE10.3, //)
107 FORMAT(7X,A6,9X,A6,9X,A6)
110 FORMAT(///,' INPUT DECK VALUES:',///,8X,' XN,XN1,KONST',///,8X,
1 'RCLOUD,CONSTP,BDYTEM,DEN,CORMAS',///,8X,
2 'SIGMA,DELT,AMPO,VMXIN,VMXOUT',///,8X,
3 'ITSTRT,ITSTOP,IDIAG,ISOADI,ITYPE,NTAPE,NMODL,ISTOR,IGRID,IS
4/',13X,'MAXTRM',///,8X,
5 'BOTBDY,TOPBDY,SIDBDY',///)
C*111 FORMAT(///,' SYMMETRIES AND BOUNDARY CONDITIONS:')
112 FORMAT(6X,'NO SYMMETRIES ASSUMED.')
113 FORMAT(6X,'FULL 2-PI, BUT SYMMETRY THRU EQUATORIAL PLANE.')
114 FORMAT(6X,'PI-SYMMETRY AND SYMMETRY THRU EQUATORIAL PLANE.')
115 FORMAT(6X,'2D RUN; NO OTHER SYMMETRIES.')
116 FORMAT(6X,'2D RUN WITH SYMMETRY THRU EQUATORIAL PLANE.')
117 FORMAT(6X,'BOUNDARY CONDITION AT BOTTOM OF GRID IS '',A6,'''
1 /,6X,'BOUNDARY CONDITION AT TOP OF GRID IS '',A6,'''
2 /,6X,'BOUNDARY CONDITION AT SIDE OF GRID IS '',A6,'''///)
135 FORMAT(///,'-------THIS RUN ASSUMES PI-SYMMETRY-------',///)
2222 FORMAT(1P10E12.5)
READ(5,100)XN,XN1,KONST
READ(5,100)RCLOUD,CONSTP,BDYTEM,DEN,CORMAS
READ(5,100)SIGMA,CIRP,AMPO,VMXIN,VMXOUT
READ(5,102)ITSTRT,ITSTOP,IDIAG,ISOADI,ITYPE,NTAPE,NMODL,IST
1 GRID,ISYM,MAXTRM
READ(5,107)BOTBDY,TOPBDY,SIDBDY
ISYMA=IABS(ISYM)
IF(ISYMA.EQ.2.OR.ISYMA.EQ.3.OR.ISYMA.EQ.9)BOTBDY=' WALL'
IF(BOTBDY.NE.' WALL'.AND. BOTBDY.NE.' FREE')BOTBDY=' DIRICH'

FOR POLYTROPES, ISOADI MUST = 3.
IF(ISOADI.EQ.2.OR.ISOADI.EQ.3.OR.ISOADI.EQ.9)WRITE(7,104)ITSTRT,ITSTOP
IF(ISOADI.EQ.1)WRITE(7,103)ITSTRT,ITSTOP,IDIAG
IF(VMXIN.LE.0.0) VMXIN=12.5
IF(VMXOUT.LE.0.0) VMXOUT=7.5
WRITE(7,110)
WRITE(7,100)XN,XN1,KONST
WRITE(7,100)RCLOUD,CONSTP,BDYTEM,DEN,CORMAS
WRITE(7,100)SIGMA,CIRP,AMPO,VMXIN,VMXOUT
WRITE(7,102)ITSTRT,ITSTOP,IDIAG,ISOADI,ITYPE,NTAPE,NMODL,IST
GRID,ISYM,MAXTRM
WRITE(7,107)BOTBDY,TOPBDY,SIDBDY
IF(ISYMA.EQ.1)WRITE(7,112)
IF(ISYMA.EQ.2)WRITE(7,113)
IF(ISYMA.EQ.3)WRITE(7,114)
IF(ISYMA.EQ.8)WRITE(7,115)
IF(ISYMA.EQ.9)WRITE(7,116)
WRITE(7,117)BOTBDY,TOPBDY,SIDBDY
IF(ISOADI.EQ.3)WRITE(7,106)XN,XN1,KONST

C.. INITIALIZE FOLLOWING PARAMETERS FOR ALL RUNS.
IF(IABS(ISYM).GT.5)LMAX=1
PI = 3.1415926535897E0
GRAV=667.32
IF(ISQADI.EQ.3)GRAV=1.0
DTHETA=2.0*PI/LMAX
   IF(IABS(ISYM).EQ.3)DTHETA=0.5*DTHETA
   IF(IABS(ISYM).EQ.3)WRITE(7,135)
   GAMMA=XN1
   CVHEAT=CURLYR/(XMÜ* (GAMMA-1.0))
   THETA=-0.5*DTHETA
   DO 4 L=1,LMAX
      THETA=THETA+DTHETA
      COSIGN(L) =COS( THETA)
      SIGN(L)=SIN(THETA)
   4 CONTINUE
   IF YOU ARE READING IN OLD MODEL FROM DISK (ITYPE=1), THEN
   SOME OF THE FOLLOWING PARAMETERS MAY BE RESET AT THAT TIME.
   TIME=0.0
   DENEX=1.0E-7*DEN
   TMASS=0.0
   ENEW=0.0
   ELOST=0.0
   EDIF=0.0
   PHICHK=0.0
   KLOCAT=0
   MLOSTR=0.0
   MLOSTZ=0.0
   JLOSTZ=0.0
   JLOSTR=0.0
   NM=NM0DL-1
   IF(NM.LE.0) GO TO 435
   READ(NTAPE,2222)U,W,JN,RHO,ROF3N,ZOF3N,A1NEWR,A1NEWZ,
   IDELT,TIME,ELOST
   435 READ(NTAPE,2222)U,W,JN,RHO,ROF3N,ZOF3N,A1NEWR,A1NEWZ,
C* 1DELT, TIME, ELOST
READ(9) S, T, A, RHO, ROF3N, ZOF3N, A1NEWR, A1NEWZ, DELT, TIME, MLOSTR, 
1 MLOSTZ, JLOSTR, JLOSTZ
440 CONTINUE
C* THEN CALCULATE R, Z, RHF, AND ZHF
DELR=ROF3N
DELZ=ZOF3N
R(3)=DELR
R(2)=0.0
R(1)=-R(3)
R(4)=2.0*DELR
DO 448 J=5,JMAX1
DELR=A1NEWR*DELR
R(J)=R(J-1)+DELR
448 CONTINUE
R(JMAX2) = R(JMAX1)+DELR
RCLOUD = R(JMAX1)
Z(3)=DELZ
Z(2)=0.0
Z(1)=-Z(3)
Z(4)=2.0*DELZ
DO 450 K=5,KMAX1
DELZ=A1NEWZ*DELZ
Z(K)=Z(K-1)+DELZ
450 CONTINUE
Z(KMAX2) = Z(KMAX1) + DELZ
DO 452 J=1,JMAX1
452 RHF(J)=0.5*(R(J)+R(J+1))
DO 454 K=1,KMAX1
454 ZHF(K)=0.5*(Z(K)+Z(K+1))
IF(ITYPE.EQ.1) GO TO 550
C.. FINISH SETTING UP INITIAL MODEL.
C* IF(ISOADI.EQ.3)CALL IZUMI(ICALL,DUMMY,DUMMY)
ICALL = ITYPE + 1
IF(ISOADI.EQ.3.AND.ITYPE.EQ.2)CALL IZUMI(ICALL,SIGMA,AM
IF(ISOADI.EQ.3.AND.ITYPE.EQ.3)CALL IZUMI(ICALL,SIGMA,AM
IF(ISOADI.EQ.3.AND.ITYPE.EQ.4)CALL IZUMI(ICALL,SIGMA,AM
ICALL = ITYPE + 1
IF(ISOADI.EQ.3.AND.ITYPE.EQ.5)CALL IZUMI(ICALL,SIGMA,AM
C AND SET UP RO, ZO, RHFO, ZHFO, AND G AND H.
C* DO 458 J=1,JMAX1
C* RO(J)=R(J)
C* RHFO(J)=RHF(J)
C* G(J)=0.0
C*458 CONTINUE
C* RO(JMAX2)=R(JMAX2)
C* G(JMAX2)=0.0
C* ZO(KMAX2)=Z(KMAX2)
C* H(KMAX2)=0.0
C* DO 460 K=1,KMAX1
C* ZO(K)=Z(K)
C* ZHFO(K)=ZHF(K)
C*   H(K)=0.0
C*460 CONTINUE
C
C
C 550 CONTINUE
C... FROM RHO, FIND P AND EPS FOR THE POLYTROPE.
DO 560 L=1,LMAX
   DO 560 K=1,KMAX2
      DO 560 J=1,JMAX2
         C* CV(J,K,L)=CVHEAT
         IF(ISOADI.NE.1)GO TO 554
         C* P(J,K,L)=EPS(J,K,L)*(GAMMA-1.0)*RHO(J,K,L)
            GO TO 560
   554 CONTINUE
   P(J,K,L)=KONST*RHO(J,K,L)**XN1
   C* EPS(J,K,L) = P(J,K,L)/(RHO(J,K,L)*(GAMMA-1.0))
   560 CONTINUE
C... FROM S, T, A, AND RHO FIND U, W, AND VT
CALL VEL
   RHFINV(2) = 1.0/RHF(2)**2
   DO 565 J=3,JMAX1
      JM=J-1
      RD(J)=RHF(J)**2-RHF(JM)**2
      RDM(J)=R(J)**2-RHF(JM)**2
      RDP(J)=RHF(J)**2-R(J)**2
      RINV(J)=2.0/(RHF(J)**2+RHF(JM)**2)
      RHFINV(J) = 1.0/RHF(J)**2
   565 CONTINUE
   DO 570 K=2,KMAX1
      KM=K-1
      ZD(K)=ZHF(K)-ZHF(KM)
      ZDM(K)=Z(K)-ZHF(KM)
      ZDP(K)=ZHF(K)-Z(K)
   570 CONTINUE
C... FIND POTENTIALS FOR THE MODEL.
   NZERO=0
   HEIGHT=8
   CALL SETBDY(NZERO,ISYM)
   ETEDGE=0.0
   CALL BDYGEN(MAXTRM,ISYM,REDGE)
   CALL POT3(HEIGHT,NZERO,ISYM)
RETURN
END
SUBROUTINE IZUMI(ICALL,SIGMA,AMPO)
C
PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1                        KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2                        LMAX=64)
C
COMMON /BLOK6/   DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAV
COMMON /BLOK7/   RCLASS,CONSP,DELT,BDYTEM,DEN,TIME,CORMAS
C* COMMON /GRID/
C*   R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
C*   1                                    G(JMAX2),H(KMAX2),ROF3N,ZOF3N,ALNEWR,ALNEWZ

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
COMMON /OLD/  RO(JMAX2), ZO(KMAX2), RHFO(JMAX1), ZHFO(KMAX1), JMAX, KMAX, JMAX1, KMAX1
COMMON /GRID/  R(JMAX2), Z(KMAX2), RHF(JMAX2), ZHF(KMAX1), JMAX2, KMAX2, JMAX1, KMAX1
REAL  JN
COMMON /EOM/  S(JMAX2,KMAX2,LMAX), T(JMAX2,KMAX2,LMAX), A(JMAX2,KMAX2,LMAX), U(JMAX2,KMAX2,LMAX), W(JMAX2,KMAX2,LMAX), JN(JMAX2,KMAX2,LMAX), OMEGA(JMAX2,KMAX2,LMAX), VT(JMAX2,KMAX2,LMAX)
COMMON /STATES/  P(JMAX2,KMAX2,LMAX)
COMMON /POIS/  PHI(JMAX2,KMAX2,LMAX), RHO(JMAX2,KMAX2,LMAX)
COMMON /CENPOI/  PHIC(JMAX2,KMAX2), PTMASS
REAL  KONST
COMMON /PTROPE/  XN, XN1, KONST, XNHYD
COMMON /FREEZ/  DEX, CIRP
COMMON /NORMAL/  CIRP
COMMON /JETADD/  TOVERW
DOUBLE PRECISION  DSEED, RPIMAX, ZZMAX, RATIO
DOUBLE PRECISION  DENNY, ANGGY
COMMON /FISS/  DENNY(JMAX,KMAX), ANGGY(JMAX,KMAX), JREQ, KZPO
DIMENSION  RHFINV(JMAX2)
DIMENSION  RHOGRD(JMAX2,KMAX2), VGRAD(JMAX2), OMHF(JMAX2)
IF(DDD.LE.DENCEN)GO TO 222
DENCEN = DDD
KZMAX = K
JRMAX = J

222 CONTINUE
DEN = DENCEN
WRITE(7,*)' DEN=',DEN,' THIS IS IMPORTANT'
PTMASS = RATIO
DENEX = 1.0E-7 *DENCEN
DDELR = RPIMAX/FLOAT(JMAX-1)
DDELZ = ZZMAX/FLOAT(KMAX-1)

C FIGURING OUT CIRP FROM ANGGY.
CIRP=2*3.14159265358979323846*(DDELR/2.0)**2/ANGGY(2,2)
WRITE(7,*)' THIS IS IMPORTANT'
WRITE(7,*)' CIRP=',CIRP
A1NEWZ=1.0
ROF3N=DDELR
ZOF3N=DDELZ
A1NEW=1.0

C ................FOLLOWING ARE DUMMY VALUES FOR IZUMI'S MODEL.
PINDEX = XN
CON2 = PTMASS
RRR2 = RPIMAX
OMCEN = ZZMAX
TOVERW = 1.0
JREQ = JRMAX
KZPOL = KMAX

WRITE(7,136)PINDEX,CON2,RRR2,OMCEN,DENCEN,TOVERW,DDELR,
1 DDELZ,A1NEWZ,JREQ,KZPOL
136 FORMAT(/,5X,'PINDEX = ',1PE11.3,/,5X,'PTMASS = ',1PE11.3,/,1
1 5X,'RPIMAX = ',1PE11.3,/,5X,'ZZMAX = ',1PE11.3,/,5X,'DENMAX
2 1PE11.3,/,5X,'TOVERW = ',1PE11.3,/,5X,'DEL = ',1PE11.3,5X,
3 'DELZ = ',1PE11.3,/,5X,'A1NEWZ = ',1PE11.3,/,5X,'JMAX = ',
4 15,/,5X,'KMAX = ',I5,/,/) RETURN

200 IF(ICALL.NE.2)GO TO 300
RETURN

300 CONTINUE
C ZERO OUT U, W, JN, AND RHO ARRAYS.
DO 310 L=1,LMAX
DO 310 K=1,KMAX2
DO 310 J=1,JMAX2
U(J,K,L)=0.0
W(J,K,L)=0.0
JN(J,K,L)=0.0
RHO(J,K,L)=0.0
310 CONTINUE
   IF (ICALL .NE. 3) GO TO 400

   IF (ICALL = 3, SET UP UNPERTURBED MODEL (i.e., AXISYMMETRIC).
   DO 312 K = 1, KMAX
      KD = K
   DO 312 J = 1, JMAX
      JD = J
   DO 312 L = 1, LMAX
      RHO(JD, KD, L) = DENNY(J, K)
      JN(JD, KD, L) = ANGGY(J, K)
   312 CONTINUE
   GO TO 500

400 CONTINUE
   IF (ICALL .NE. 4) GO TO 450
   GO TO 500

450 CONTINUE
   IF (ICALL .NE. 5) GO TO 800
   IF (ICALL = 5, PERTURB MODEL WITH RANDOM DENSITY PERTURBATION H
   MAXIMUM AMPLITUDE OF AMP0.
   DSEED = TOVERW
   IF (DSEED .LE. 1. D-8) DSEED = 1. DO
   DSEED = 1.0/DSEED
   WRITE (7, 1001) AMP0, DSEED
1001 FORMAT (5X, 'GOT TO RANDOM, AMP0 =', 1PE15.4, /
1 5X, 'INITIAL SEED TO GGUBFS = ', 1PD15.4)
   DO 462 L = 1, LMAX
   DO 462 K = 2, KMAX
      KD = K
   DO 462 J = 2, JMAX
      JD = J
      JN(JD, KD, L) = ANGGY(J, K)
      C GGUBFS IS AN IMSL RANDOM NUMBER FUNCTION.
      C PERT = GGUBFS(DSEED)
      C RAN IS AN FPS RANDOM NUMBER FUNCTION.
      PERT = RAN(DSEED)
      PERT = AMP0 * (2.0 * PERT - 1.0)
      RHO(JD, KD, L) = (1.0 + PERT) * DENNY(J, K)
   462 CONTINUE
   GO TO 500

800 CONTINUE
   IF (ICALL .NE. 6) GO TO 500
   WRITE (7, 1002) AMP0, SIGMA
1002 FORMAT (' PERTURBING INITIAL MODEL STRAIGHT AMP0 =', 1PE10.3, ',
1     MODE =', 0PF6.3)
   IF (ICALL = 6, PERTURB MODEL WITH STRAIGHT M = INT(SIGMA) PERTURB
   OF AMPLITUDE AMP0. (ACTUALLY, RAMP AMPLITUDE UP TO AMP0)
THETA=-DTHETA
EM=SIGMA
DO 805 L=1,LMAX
THETA=THETA+DTHETA
EMT=EM*THETA
PERT=AMP0*COS(EMT)
DO 805 K=2,KMAX
KD=K
DO 805 J=2,JMAX
JD=J
JN(JD,KD,L)=ANGGY(J,K)
IF(JD.LE.3)PF=0.0
IF(JD.GT.3.AND.JD.LT.7)PF=0.25*FLOAT(JD-3)
IF(JD.GE.7)PF=1.0
RHO(JD,KD,L)=(1.0+PF*PERT)*DENNY(J,K)
805 CONTINUE

C
C
C
C
C
C
C

500 CONTINUE
C
C... ON 12/3/85, CHANGED LIMITS OF LOOP FROM MAX2 TO MAX1 BECAUSE
C... OUTERMOST ZONES REALLY SHOULDN'T NEED ANY DENSITY.
DCHEK = 1.0E-7 * DENCEN
DO 525 L=1,LMAX
DO 525 K=2,KMAX1
DO 525 J=2,JMAX1
IF(RHO(J,K,L).GE.DCHEK)GO TO 525
RHO(J,K,L)=DENEX
525 CONTINUE
C
CCON 12/18/86 AT LSU, ADDED INITIALIZATION OF EPS HERE IN ORD
C TO ALLOW GAMMA.NE.(1+1/XN1) ADIABATIC EVOLUTIONS.
C
C... INITIALIZE EPS:
GAMMA = XNHKD
C* DO 506 L=1,LMAX
C* DO 506 K=2,KMAX1
C* DO 506 J=2,JMAX1
C* PP0 = KONST*RHO(J,K,L)**XN1
C* EPS(J,K,L) = PP0/(RHO(J,K,L) * (GAMMA - 1.0))
506 CONTINUE
C
C MAKE SURE VALUES OF VARIABLES ON BOARDERS OF GRID ARE CORREC
C INITIALIZED.
DO 520 L=1,LMAX
DO 517 K=2,KMAX2
RHO(1,K,L)=RHO(2,K,L)
C* EPS(1,K,L)=EPS(2,K,L)
JN(1,K,L)=JN(2,K,L)
U(1,K,L)=-U(3,K,L)
517 W(1,K,L)=W(3,K,L)
DO 519 J=1,JMAX2
RHO(J,1,L)=RHO(J,2,L)
C* EPS(J,1,L)=EPS(J,2,L)
JN(J,1,L)=JN(J,2,L)
U(J,1,L)=U(J,3,L)
519 W(J,1,L)=-W(J,3,L)
520 CONTINUE
C
DO 5114 J=1,JMAX1
DO 5114 K=2,KMAX,5
KD=K+4
WRITE(7,134)(RHO(J,KK,1),KK=K,KD)
5114 CONTINUE
DO 5115 J=1,JMAX1
DO 5115 K=2,KMAX,5
KD=K+4
WRITE(7,134)(JN(J,KK,1),KK=K,KD)
5115 CONTINUE
DO 952 J=1,JMAX
DO 951 K=1,KMAX
DO 950 L=1,LMAX
A(J,K,L)=RHO(J,K,L)*JN(J,K,L)
950 CONTINUE
951 CONTINUE
952 CONTINUE
RETURN
END
SUBROUTINE GEM(ISYM,BOTBDY,TOPBDY,SIDBDY)
C FOR THE FULLY SECOND ORDER DIFFERENCE IN EVERYTHING.
PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)
C
C COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SSIGN(LMAX),PI,GRAV
C COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
1 ROF3N,ZOF3N,ANEWR,ANEWZ
C COMMON /GEX/ GEMS,GEMR,GEMZ(JMAX1),VOLMI(JMAX1),DT(JMAX1),
1 RHF(JMAX1),RHFJ2(JMAX1),DR,DZ,DELTHI,RHFJ3(JMAX1)
CHARACTER*6 BOTBDY, TOPBDY, SIDBDY
C IT IS ASSUMED THAT THE GRID IS UNIFORM IN Z AND R AND NOT MOVING
C OR Z. THERE SHOULD NEVER BE ANY REASON FOR
C NONUNIFORMITY IN THETA. IF Z IS NOT UNIFORM THE GEM ARRAYS WILL H
C TO VARY IN K ALSO AND BE TWO DIMENSIONAL ARRAYS. ALSO IF THE G
C IN R OR Z IS ALLOWED TO MOVE THIS WILL HAVE TO BE UPDATED EVERY
C TIMESTEP. NOW IT IS CALLED ONCE AND FOR ALL.
DELTHI=0.5/DTHETA
DZ=Z(5)-Z(4)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DR=R(5)-R(4)
C
DZPI=DZ*PI/(2*LMAX)
DZPI2=DZ*DTHETA/2.0
GEMS=DZ*DR
GEMR=DZ*DTHETA
DO 10 J=2,JMAX1
JP=J+1
R2DIF=R(JP)**2-R(J)**2
VOLMI(J)=1.0/(DZPI*R2DIF)
GEMZ(J)=DTHETA/2.0*R2DIF
DT(J)=RHF(J)*DTHETA
RHF3(J)=1.0/(RHF(J)**3)
10 CONTINUE
C
RHF2(J)=1.0/RHF(J)**2
DO 501 J=2,JMAX1
RHFI(J)=1.0/RHF(J)
RHF2(J)=RHFI(J)/RHF(J)
501 CONTINUE
C GEMR ARE MISSING AN R(J)* FACTOR EACH
C SINCE THIS IS A FUNCTION OF VELOCITY IT CAN NOT BE COMPUTED JUST
C ONCE.
CALL SETUP2
RETURN
END
SUBROUTINE SETUP2

C
PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)
C
C* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
REAL  JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
C A AND RHO ARE ZEROED HERE ONCE SO NOTHING WILL BE IN THESE ZONES
C HOPEFULLY IT WILL STAY THAT WAY WITH A COMPLETELY ABSORBING EDGE
C BUT MLOSTZ AND MLOSTR WILL KEEP TRACK OF MASS LOST IN EOC AND
C JLOSTZ AND JLOSTR WILL KEEP TRACK OF ANGULAR MOMENTUM LOST IN EOC
DMAX=RHO(JMAX2,KMAX2,1)
DO 2 L=1,LMAX
DO 1 K=1,KMAX1
A(JMAX1,K,L)=0.0
A(JMAX2,K,L)=0.0
S(JMAX1,K,L)=0.0
S(JMAX2,K,L)=0.0
T(JMAX1,K,L)=0.0
T(JMAX2,K,L)=0.0
RHO(JMAX1,K,L)=DMAX*1.0E-07
RHO(JMAX2,K,L)=DMAX*1.0E-07
1 CONTINUE
2 CONTINUE
DO 4 L=1,LMAX
DO 3 J=2,JMAX
A(J,KMAX1,L)=0.0
A(J,KMAX2,L)=0.0
S(J,KMAX1,L)=0.0
S(J,KMAX2,L)=0.0
T(J,KMAX1,L)=0.0
T(J,KMAX2,L)=0.0
RHO(J,KMAX1,L)=DMAX*1.0E-07
RHO(J,KMAX2,L)=DMAX*1.0E-07
3 CONTINUE
4 CONTINUE
RETURN
END
C* IN THE MAIN LOOP VEL IS ONLY CALLED ONCE.
C* THIS PROGRAM IS DESIGNED TO HANDLE ROTATING POLYTOPES WITH AN
C* EULARIAN GRID WHICH ROTATES BUT DOES NOT MOVE IN R AND Z.
C* RIGHT NOW UNIFORM ROTATION OF THE GRID IS DISABLED.
C* SYMMETRY ABOUT THE EQUATORIAL PLANE IS ASSUMED BUT SYMMET
C* ABOUT THE ROTATION AXIS IS EITHER PI SYMMETRY OR 2PI, NO SYMMET
C* ABOUT THE ROTATION AXIS.
C* THIS CONTINUES ALLREADY STARTED MODELS.
C* SECOND ORDER MODELS ARE RUN.
C* JTV AND OMEGA STILL NEED TO BE REMOVED.
C* AND WRITTEN OUT AS S, T, A, RHO, ROF3N, ZOF3N, DELT, TIME, MLOSTR, MLOST
C* JLOSTR, JLOSTZ.

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
  1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
  2 LMAX=64)

COMMON /STEP/ ICNT
COMMON /BLOK6/ DTHETA, COSIGN(LMAX), SIGN(LMAX), PI, GRAV
COMMON /BLOK7/ RCLoud, CONSTP, DELT, BDYTEM, DEN, TIME, CORMAS
COMMON /GRID/ R(JMAX2), Z(KMAX2), ROF3N, ZOF3N, AlNEWR, AlNEWZ
REAL JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX), T(JMAX2,KMAX2,LMAX),
  1 A(JMAX2,KMAX2,LMAX), U(JMAX2,KMAX2,LMAX),
  2 W(JMAX2,KMAX2,LMAX), JN(JMAX2,KMAX2,LMAX),
  3 OMEGA(JMAX2,KMAX2,LMAX), VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX), RHO(JMAX2,KMAX2,LMAX)
COMMON /INSIDE/ TMASS, ENEW, ELOST, EDIF, PHICBK, KLOCAT
COMMON /TIMST/ INDX, ISOADI, ALLOW, ITSTEP
CHARACTER*6 BOTBDY, TOPBDY, SIDBDY
INTEGER MINUS1, ZERO, ONE, TWO, THREE, FOUR, FIVE, EIGHT, FL5, FIFTY,
  1 LENGL, LENG2
C LENG2=8*(4*JMAX2*KMAX2*LMAX+10)=1253456
OPEN(UNIT=7, FILE=' DIAG07, LENGTH=132', STATUS=' NEW')
OPEN(UNIT=9, FILE=' MOD09, LENGTH=1253456', FORM=' UNFORMATTED',
  1 STATUS=' OLD')
OPEN(UNIT=12, FILE=' MOD12, LENGTH=120', STATUS=' NEW')
OPEN(UNIT=13, FILE=' MOD13, LENGTH=1253456', FORM=' UNFORMATTED',
  1 STATUS=' NEW')
OPEN(UNIT=25, FILE=' COEF25, LENGTH=120', STATUS=' NEW')
MINUS1=-1
ZERO=0
ONE=1
TWO=2
THREE=3
FOUR=4
FIVE=5
EIGHT=8
FL5=15
FIFTY=50

ISDIAG=5
INDX=0
CALL SETUP(ITSTRT, ITSTOP, IDIAG, ISOADI, ISTOR, ITYPE, ISYM, MAXTR)
CALL GEM(ISYM, BOTBDY, TOPBDY, SIDBDY)
C WHAT IS THE PURPOSE OF THE BELOW IF STATEMENT.
IF(RHO(JMAX2, KMAX1, 1).EQ.0.0) RHO(JMAX2, KMAX1, 1) = 1.0
CALL RITE(THREE, MINUS1, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
ITSTOP=ITSTOP-1
DO 90 ITSTEP=ITSTRT, ITSTOP
   ICNT=ITSTEP
   INDX=ITSTEP-ITSTRT
   IHEAD=0
   IF(MOD(INDX, IDIAG).EQ.0) IHEAD=-1
   IPRINT=0
   IF(MOD(INDX, IDIAG).EQ.0) IPRINT=1
   CALL DELTA
   CALL SOST(ISYM, BOTBDY, TOPBDY, SIDBDY)
   CALL SOA(ISYM, BOTBDY, TOPBDY, SIDBDY)
   CALL VIS
   CALL VEL
   CALL HYDROS(ISYM, BOTBDY, TOPBDY, SIDBDY)
   CALL EOC(ISYM, BOTBDY, TOPBDY, SIDBDY)
   CALL VEL
   IF(MOD(INDX, IDIAG).NE.0) GO TO 47
   C+ DO 44 JJJ=2,32
   C+ CALL RITE(ONE, ONE, JJJ, JJJ, ONE, TWO, KMAX2, ONE, ONE, ONE, ONE)
   C+ 44 CONTINUE
   C+ 47 CONTINUE
   C+ IF(MOD(INDX, ISDIAG).EQ.0) CALL RITE(FOUR, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
   C+ 1 CALL STATE(THREE, ZERO, ZERO, ZERO)
   REDGE=R(JMAX1)
   CALL BDYGEN(MAXTRM, ISYM, REDGE)
   CALL P0T3(EIGHT, IPRINT, ISYM)
   IF(MOD(INDX, FIFTY).EQ.0) CALL RITE(FIVE, ISYM, ICNT, ONE, ONE, ONE, ONE, ONE, ONE, ONE)
   CALL RITE(ZERO, IHEAD, ITSTEP, INDX, ISYM, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE, IS
   C FOR STORING INTERMEDIATE MODELS.
   IF(INDX.EQ.0) GO TO 40
   IF(MOD(INDX, ISTOR).EQ.0) CALL RITE(TWO, MINUS1, ITSTEP, ITS
   1 ITSTOP, ISTOR, ONE, ONE, ONE, ONE, ONE
   40 CONTINUE
   C END OF INTERMEDIATE STORE.
   90 CONTINUE
   ITSTEP=ITSTOP+1
   CALL RITE(FIVE, ISYM, ITSTEP, ONE, ONE, ONE, ONE, ONE, ONE, ONE
   C+ CALL RITE(THREE, MINUS1, ONE, ONE, ONE, ONE, ONE, ONE, ONE, ONE
   C+ CALL RITE(TWO, MINUS1, ITSTEP, ITSTRT, ITSTOP, ISTOR, ONE, ONE, ONE, ONE, ONE, ONE
CLOSE(7)
CLOSE(9)
CLOSE(12)
CLOSE(13)
CLOSE(25)
STOP
END

SUBROUTINE RITE(IWHAT, IHEAD, JST, JSP, JSK, KST, KSP, KSK, LST, LSP, IWHAT = 1 PRINTS ALL VARIABLES OUT IN 1PE12.4 FORMAT.
= 0 BRIEF DIAGNOSTICS (2 LINES) ONLY AND COEF FILE OF
= 2 STORES THIS MODEL.
= 5 BRIEF SUMMARY, MOST USEFUL.
IHEAD = NEGATIVE SKIPS TO A NEW PAGE; IHEAD.GE.0 DOES NOT.
ABS(IHEAD) = 1 PRINTS HEADING ONCE PER CALL TO SUBROUTINE.

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)

COMMON /BLOK6/ DTHETA, COSIGN(LMAX), SIGN(LMAX), PI, GRAV
COMMON /BLOK7/ R CLOUD, CONSTP, DELT, BDEM, DEN, TIME, CORMAS
COMMON /NORMAL/ CIRP
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX2), ZHF(KMAX1),
1 ROF3N, ZOF3N, AINEWR, AINNEWZ
REAL
COMMON /EOM/ S(JMAX2, KMAX2, LMAX), T(JMAX2, KMAX2, LMAX),
1 A(JMAX2, KMAX2, LMAX), U(JMAX2, KMAX2, LMAX),
2 W(JMAX2, KMAX2, LMAX), JN(JMAX2, KMAX2, LMAX),
3 OMEGA(JMAX2, KMAX2, LMAX), VT(JMAX2, KMAX2, LMAX)
COMMON /STATES/ P(JMAX2, KMAX2, LMAX)
1 EPS(JMAX2, KMAX2, LMAX)

COMMON /TEMCV/ CVHEAT
COMMON /POIS/ PHI(JMAX2, KMAX2, LMAX), RHO(JMAX2, KMAX2, LMAX)
COMMON /USE2/ RD(JMAX2), RDOLD(JMAX2), RDB(JMAX2),
1 RNV(JMAX2), R3(JMAX2), RDM(JMAX2),
2 RDP(JMAX2), GRADR(JMAX2), ZD(KMAX2),
3 ZDOLD(KMAX2), ZDB(KMAX2), ZDM(KMAX2),
4 ZDP(KMAX2), GRADZ(KMAX2)
COMMON /INSIDE/ TMASS, ENEW, ELOST, EDIF, PHICKEK, KLOCAT
COMMON /IRITE/ EGOLD, EKOLD, PDVOLD
COMMON /COEFS/ COEF(JMAX, LMAX, 2)
REAL
COMMON /ANGMOM/ JLOSTR, JLOSTZ, MLOSTR, MLOSTZ
COMMON /SUMRY/ EM(JMAX1), ENRTIA(JMAX1), EJ(JMAX1),
1 EROT(JMAX1), EG(JMAX1), BETA(JMAX1)
COMMON /ROTFRM/ OMGFRM
REAL JNINER
DIMENSION RO(JMAX2), ZO(KMAX2), RHFO(JMAX2), ZHFO(KMAX1)
EQUIVALENCE (RO(1), R(1))
EQUIVALENCE (ZO(1), Z(1))
EQUIVALENCE (RHFO(1), RHF(1))
EQUIVALENCE (ZHFO(1),ZHF(1))
DIMENSION WORK(JMAX,6)
EQUIVALENCE (WORK(1,1),EM(1))

C**OMGFRM SET HERE BECAUSE ON A CONTINUED MODEL THIS CARD IS NOT RE
C**FROM DATASET UNIT=5.
OMGFRM=0.0
100 FORMAT(1H1)
101 FORMAT(///,' J K L ',5X,'JN',9X,'OMEGA',8X,'RHO',9X,'PHI',1
           5X,'JN-INERTIAL',1X,'OM-INERTIAL',7X,'U',11X,'W',//)
102 FORMAT(3I3,1P8E12.4)
103 FORMAT(///)
104 FORMAT( ' TSTEP',3X,'TIME',8X,'DELT',6X,'ETOT/JS',4X,'TEGRV/1,
           2X,'EKIN/RKIN',3X,'ALPHA/CD',3X,'EDIF/DMAX',3X,'ELOST/2
           6X,'TMASS',6X,'PHICHK',4X,'K' ,//)
105 FORMAT(I5,1P10E12.4,I4)
106 FORMAT(5X,1PE12.4,'CIRPS',6X,1P5E12.4,I4,2I3,2X,1P2E12.4//)
107 FORMAT(10X,'JLOST =',1P2E12.4,41X,'S MODEL FROM TIME STEP NUMBER' ,I5,'S HAS
           2' S BEEN STORED ON DISK. IT IS STORED AS S',1/' S MODEL
           3',I2,' ON TAPE UNIT',I3,'.',8X,'S',1/' S',41X,'S',//
           4' SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS',///)
110 FORMAT(IH1,///,'  SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS',41X,'S',1//,
           ' S MODEL FROM TIME STEP NUMBER',15,' HAS
           2' S BEEN STORED ON DISK. IT IS STORED AS S',1/' S MODEL
           3',I2,' ON TAPE UNIT',I3,'.',8X,'S',1/' S',41X,'S',//
           4' SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS',///)
111 FORMAT(1P5E13.5,/,1P4E13.5)
112 FORMAT(1P6E13.5)
113 FORMAT(7I10)
114 FORMAT(1P6E13.5)
115 FORMAT(1P10E12.5)
IF(IWHAT.NE.1) GO TO 50

C IWHAT = 1,
C PRINT OUT DETAILED VARIABLE ARRAY VALUES.
C
IF(IHEAD.LT.0) WRITE(6,100)
IF((IHEAD*IHEAD).EQ.1) WRITE(6,101)
IF(IABS(IHEAD).NE.1) WRITE(6,103)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 40 L=LST,LSP,LSK
DO 40 K=KST,KSP,KSK
DO 40 J=JST,JSP,JSK
OMINER = OMEGA(J,K,L) + OMGFRM
JNINER = OMINER*RHF(J)**2
WRITE(6,102) J,K,L,JN(J,K,L),OMEGA(J,K,L),RHO(J,K,L),PHI(J,K)
1 JNINER,OMINER,U(J,K,L),W(J,K,L)
40 CONTINUE
GO TO 999
50 CONTINUE
ISYMA=IABS(JSK)
IF(IWHAT.NE.0) GO TO 95
ccccc
C WHAT = 0 ,
C C
C PRINT OUT BRIEF SUMMARY OF GLOBAL VARIABLES,
C AND (AS OF 3/20/85 AT LSU) WRITE FFT COEFS TO DISK.
C
ccccc
C+ IF(IHEAD.LT.0) WRITE(6,100)
C+ IF(IHEAD.GT.0) WRITE(6,103)
C+ IF(IABS(IHEAD).EQ.1) WRITE(6,104)
C* WIRD NOW PUT IN SUBROUTINE DELTA
TCIRP=TIME/CIRP
DO 55 J=2,JMAX1
JP=J+1
RD(J)=R(JP)**2-R(J)**2
55 CONTINUE
DO 57 K=2,KMAX1
KP=K+1
ZD(K)=Z(KP)-Z(K)
57 CONTINUE
C.. .  FIND EGRAV,TOTJ, EROT.
AREA=0.5*DTHETA
EEG=0.0
TOTJ=0.0
ER=0.0
DO 60 L=1,LMAX
DO 60 K=2,KMAX
AREA2=AREA* ZD( K)
DO 60 J=2,JMAX
VOL=RD(J)*AREA2
EEG=EEG+ 0.5*VOL*PHI(J,K,L)*RHO(J,K,L)
TOTJ=TOTJ + A(J,K,L)*VOL
ER=ER+0.5*VOL*A(J,K,L)*OMEGA(J,K,L)
60 CONTINUE
C
DO 65 J=3,JMAX1
JM=J-1
RD(J)=RHFO(J)**2-RHFO(JM)**2
RINV(J)=1.0/RO(J)**2
65 CONTINUE
DO 67 K=2,KMAX1
   KM=K-1
   ZD(K)=ZHFO(K)-ZHFO(KM)
67 CONTINUE
   ZD(2)=0.5*ZD(2)
C... FIND RZKIN
   RJK=0.0
   DO 70 L=1,LMAX
      DO 70 K=2,KMAX1
         AREA1=AREA*ZD(K)
         DO 70 J=3,JMAX1
            VOL=RD(J)*AREA1
            RJK=RJK+0.5*VOL*(U(J,K,L)*S(J,K,L) + W(J,K,L)*T(J,K,L))
70 CONTINUE
   AREA1=PI*RHF0(2)**2
   DO 75 K=2,KMAX1
      VOL=AREA1*ZD(K)
      RJK=RJK+0.5*VOL*W(2,K,1)*T(2,K,1)
75 CONTINUE
C
C... MULTIPLY INTEGRAL PROPERTIES BY FACTOR, DEPENDING ON
C SYMMETRY BEING USED.
   IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)FACTOR=1.0
   IF(ISYMA.EQ.2.OR.ISYMA.EQ.9)FACTOR=2.0
   IF(ISYMA.EQ.3)FACTOR=4.0
   EGRAV=FACTOR*EEG
   TOTJN=FACTOR*TOTJ
   EEROT=FACTOR*ER
   RZKIN=FACTOR*RJK
   EKIN=EEROT+RZKIN
   JRHO=2
   KRHO=2
   LRHO=1
   RHOMAX=RHO(2,2,1)
   DO 80 I=1,LMAX
      DO 80 K=2,KMAX
         DO 80 J=2,JMAX
            IF(RHO(J,K,L).LE.RHOMAX)GO TO 80
            RHOMAX=RHO(J,K,L)
            JRHO=J
            KRHO=K
            LRHO=L
80 CONTINUE
   DMAX=RHOMAX
   JD=JRHO
   KD=KRHO
   LD=LRHO
   CD=RHO(2,2,1)
   PDV=0.0
   ETOT=EGRAV+EKIN+ENEW+ELOST-PDV
   ECHECK=0.0
   IF(JSP.GE.0) GO TO 87
   ECHECK=(EKin-EKOLD + EDIF)/(EGOLD-EGRAV + PDV-PDVOLD)-1.0
87 EGOLD=EGRAV

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
PDVOLD=PDV
EKOLD=EKIN

C... INSERTS BY JOEL AT LSU 5/16/83 TO WRITE ALPHA, BETA, ETC.
C
EGG=ABS(EGRAV)
ETOT=ETOT/EGG
EKIN=EKIN/EGG
ALPHA=ENEW/EGG
EDD=EDIF/EGG
ELL=ELOST/EGG
BBETA=EEROT/EGG

C... STOP INSERT.
C
C+ WRITE(6,105) JST,TIME,DELT,ETOT,EGRAV,EKIN,ALPHA,EDD,ELL,TMA
C+ 1 PHICK,KLOCAT
C+ WRITE(6,106)TCIRP,TOTJN,BBETA,RZKIN,CD,DMAX,JD,KD,LD,ECHECK
C
C... INSERT BY JOEL AT LSU 4/17/85 TO WRITE ALL COEF TO DISK FILE
C
IF(MOD(JST,LSK).NE.0) GO TO 999
IF(LMAX.EQ.1) GO TO 1294
LHAF=6
WRITE(25,2001)JST,TIME,DELT,ETOT,EGRAV,EKIN,ALPHA,EDD,E
TMASS
WRITE(25,2002)TCIRP,TOTJN,BBETA,RZKIN,CD,DMAX
WRITE(25,2003)JD,KD,LD,JMAX,KMAX,LMAX,LHAF
WRITE(25,2004)(((COEF(JE,MM,III),MM=1,LHAF),111=1,2),
1 JE=2,JMAX)
1294 CONTINUE
C... STOP INSERT.
C
C+ WRITE(6,107)JLOSTR,JLOSTZ,MLOSTR,MLOSTZ
IF(LMAX.EQ.1) GO TO 999
LHAF=8
C+ WRITE(6,121)(JE,((COEF(JE,MM,III),MM=1,LHAF),III=1,2),
C+ 1 JE=2,JMAX,2)
GO TO 999
95 CONTINUE
NTAPES=12
IF(IWHAT.NE.2) GO TO 300
ITSTEP=JST

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
ISTRT=JSP
ISTOP=JSK
ISTOR=KST
INDX=ISTEP-ISTRT
NUM=INDX/ISTOR
IF(MOD(INDX,ISTOR).NE.0) NUM=NUM+1
IF(ISTEP.GT.ISTOP) GO TO 250
IF(NUM.GE.3) GO TO 999
250 CONTINUE
IF(NUM.GT.3) NUM=3
JN(1,KMAX2,1) = JLOSTR
JN(2,KMAX2,1) = JLOSTZ
JN(3,KMAX2,1) = MLOSTR
JN(4,KMAX2,1) = MLOSTZ
C* WRITE(NTAPES,2222)U,W,JN,RHO,EPS,ROF3N,ZOF3N,ALNEWR,ALNEWZ,
   C* 1DELT,TIME,ELOST
WRITE(NTAPES,2222)U,W,JN,RHO,ROF3N,ZOF3N,ALNEWR,ALNEWZ,
   1DELT,TIME,ELOST
C* WRITE(13)U,W,JN,RHO,ROF3N,ZOF3N,ALNEWR,ALNEWZ,
   C* 1DELT,TIME,ELOST
WRITE(13)U,W,JN,RHO,ROF3N,ZOF3N,ALNEWR,ALNEWZ,DELT,TIME,MLOST
C+ WRITE(6,110) ITSTEP,NUM,NTAPES
C+ WRITE(6,111) ITSTEP,NUM,NTAPES
300 CONTINUE
IF(IWHAT.NE.3) GO TO 400
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C IWAT = 3,
C PRINT OUT CURRENT GRID STRUCTURE.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C* WRITE(6,115)ROF3N,ALNEWR,CORMAS
C* WRITE(6,116) R,RHF
C* WRITE(6,117) ZOF3N,ALNEWZ
C* WRITE(6,116) Z,ZHF
400 CONTINUE
IF(IWHAT.NE.4) GO TO 450
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C IWAT = 4,
C PRINT OUT RADIAL BEHAVIOR OF RHO AND VELOCITIES AT
C A) K = 2
C B) J = JRHO
C BOTH BEING DONE AT L = IHEAD (OR L = LRHO IF IHEAD = 0).
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
WRITE(6,118)
LL=IHEAD
RHOMAX=0.0
DO 405 L=1,LMAX
DO 405 K=2,KMAX
DO 405 J=2,JMAX
IF(RHO(J,K,L).LE.RHOMAX)GO TO 405
RHOMAX=RHO(J,K,L)
JRHO=J
KRHO=K
LRHO=L
405 CONTINUE
IF(LL.EQ.0) LL=LRHO
IF(LL.LT.1.0R.LL.GT.LMAX)LL=1
WRITE(6,125)
DO 411 KK=2,KMAX1
WRITE(6,119)KK,JRHO,R(KK),RHO(KK,2,LL),PHI(KK,2,LL),OMEGA(KK 1 ,U(KK,2,LL),Z(KK),RH0(JRH0,KK,LL),0MEGA(JRH0,KK,LL), 2 ,KK,LL)
411 CONTINUE
KSTRTJ=KMAX1+1
IF(JMAX1.LE.KMAX1)GO TO 414
DO 412 KK=KSTRTJ,JMAX1
WRITE(6,119)KK, JRHO,R ( KK),RH0(KK,2,LL),PHI( KK,2,LL) , 1 OMEGA(KK,2,LL),U(KK,2,LL)
412 CONTINUE
414 CONTINUE
450 CONTINUE
C
C
IF(IWHAT.NE.5)GO TO 500

C
C
C
ISYM = IHEAD
AREA=0.5*DTHETA
ISYMA=IABS(ISYM)
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)FACTOR=1.0
IF(ISYMA.EQ.2.OR.ISYMA.EQ.9)FACTOR= 2.0
IF(ISYMA.EQ.3)FACT0R=4.0
C
C
DO 455 J=2,JMAX1
JP=J+1
RD(J)=R(JP)**2-R(J)**2
455 CONTINUE
DO 457 K=2,KMAX1
KP=K+1
ZD(K)=Z(KP)-Z(K)
457 CONTINUE
C... FIND EGRAV,TOTJ,EROT.
DO 459 J=1,JMAX1
DO 459 I=1,6
WORK(J,I)=0.0
CONTINUE
DO 462 J=2,JMAX
    AREA1=AREA*RD(J)
    RPI2=RHF(J)**2
    DO 460 L=1,LMAX
        DO 460 K=2,KMAX
            VOL=2D(K)*AREA1
            EMAS=RHO(J,K,L)*VOL
            ERTIA=EMAS*RPI2
            EM(J)=EM(J) + EMAS
            EJ(J)=EJ(J) + A(J,K,L)*VOL
            ENRTIA(J)=ENRTIA(J) + ERTIA
            EROT(J)=EROT(J) + 0.5*ERTIA*OMEGA(J,K,L)**2
            EG(J)=EG(J) + 0.5*EMAS*PHI(J,K,L)
        460 CONTINUE
    461 WORK(J,I)=FACTOR*WORK(J,I)+WORK(J-1,I)
    BETA(J)=EROT(J)/ABS(EG(J))
462 CONTINUE
C
C..................SINCE THIS SUMMARY GIVES MASS, ANGULAR MOMENTUM, ETC. 'INTER TO A GIVEN RADIUS, MUST MULTIPLY CURRENT (K,L) SHELL BY SYMM 'FACTOR' AND THEN ADD IN ALL PREVIOUS RADIAL SHELLS.

WRITE(7,140)JST,TIME
WRITE(7,141)(J,RHF(J),(WORK(J,I),1=1,6),J=2,JMAX)
500 CONTINUE
999 CONTINUE
RETURN
END
SUBROUTINE SETUP(ITSTRT,ITSTOP,IDIAG,ISOADI,ISTOR,ITYPE,ISYM
1 MAXTRM,BOTBDY,TOPBDY,SIDBDY)
C
C.. .  ITYPE TELLS WHETHER INITIAL OR READ IN MODEL.
C = -2, READ IN SANDFORD-WHITAKER-KLEIN MODEL FROM NTAPE.
C = 1  MEANS READ MODEL NUMBER NMODL FROM UNIT NTAPE.
C = 2  POLYTROPE MODEL, INITIALLY AXISYMMETRIC.
C = 3  POLYTROPE MODEL, PERTURBED W/ MACLAURIN BAR MODE.
C = 4  POLYTROPE MODEL, PERTURBED W/ RANDOM DENSITY PERTUR
C = 5  POLYTROPE, PERTURBED W/ STRAIGHT PERTURBATION.
NTAPE = TAPE UNIT FROM WHICH TO READ MODEL.
NMODL = NUMBER OF MODEL IN THAT FILE TO BE USED.

PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64, MWSAVE = 4*LMAX + 15)
C
COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAY
COMMON /BLOK7/ RCLCLOUD,CONSTP,DELT,BDYTEM,DEN,TIME,CORMAS
COMMON /NORMAL/ CIRP

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX1), ZHF(KMAX1),
1 R0F3N, Z0F3N, A1NEWR, A1NEWZ
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX2), ZHF(KMAX1),
1 IGRID
DIMENSION RO(JMAX2), ZO(KMAX2), RHFO(JMAX2), ZHFO(KMAX2)
EQUIVALENCE (RO(1), R(1))
EQUIVALENCE (ZO(1), Z(1))
EQUIVALENCE (RHFO(1), RHF(1))
EQUIVALENCE (ZHFO(1), ZHF(1))

COMMON /OLD/ R0(JMAX2), Z0(KMAX2), RHFO(JMAX1), ZHFO(KMAX1)

DIMENSION EPSS(JMAX2, KMAX2, LMAX)

REAL JN

COMMON /EOM/ S(JMAX2, KMAX2, LMAX), T(JMAX2, KMAX2, LMAX),
1 U(JMAX2, KMAX2, LMAX), W(JMAX2, KMAX2, LMAX),
2 JN(JMAX2, KMAX2, LMAX)

COMMON /STATES/ P(JMAX2, KMAX2, LMAX)

COMMON /STATES/ P(JMAX2, KMAX2, LMAX), CV(JMAX2, KMAX2, LMAX),

COMMON /POIS/ PHI(JMAX2, KMAX2, LMAX), RHO(JMAX2, KMAX2, LMAX),
DIVS(JMAX2, KMAX2), DIVT(JMAX2, KMAX2),
1 DIVA(JMAX2, KMAX2), SAVES(JMAX2, KMAX2),
2 SAVEJ(JMAX2, KMAX2), SAVEA(JMAX2, KMAX2),

COMMON /USE1/ DIVS(JMAX2, KMAX2), DIVT(JMAX2, KMAX2),
1 DIVA(JMAX2, KMAX2), SAVES(JMAX2, KMAX2),
2 SAVEJ(JMAX2, KMAX2), SAVEA(JMAX2, KMAX2),

COMMON /USE2/ RD(JMAX2), RDOLD(JMAX2), RDB(JMAX2),
1 RINV(JMAX2), RDM(JMAX2),
2 RDP(JMAX2), GRADR(JMAX2), ZD(KMAX2),
3 ZD0LD(KMAX2), ZDB(KMAX2), ZDM(KMAX2),
4 ZDP(KMAX2), GRADZ(KMAX2)

COMMON /INSIDE/ TMASS, ENEW, ELOST, EDIF, PHICHK, KLOCAT
REAL JLOSTR, JLOSTZ, MLOSTR, MLOSTZ

COMMON /ANGMOM/ JLOSTR, JLOSTZ, MLOSTZ, MLOSTZ

COMMON /SETPHI/ WSAVE(MWSAVE)

COMMON /VELMAX/ VMXIN, VMXOUT
REAL KONST

COMMON /PTROPE/ XN, XN1, KONST

COMMON /FREEZ/ DENEX
DIMENSION RHFINV(JMAX2)

CHARACTER*6 BOTBDY, TOPBDY, SIDBDY
DATA CURLYR, GAMMA, XMU/83.14, 1.666667E0, 2.0/

C

100 FORMAT(1P5E15.8)

102 FORMAT(1D15.8)
103 FORMAT(' THIS WILL BE AN ISOTHERMAL COLLAPSE STARTING AT TIMENumber', I4, ' AND', ' GOING THROUGH TIMESTEP NUMBER', I4, ' DIAGNOSTICS EVERY', I3, ' STEPS' ,///)
104 FORMAT(' THIS WILL BE AN ADIABATIC COLLAPSE STARTING AT TIMENumber', I4, ' AND', ' GOING THROUGH TIMESTEP NUMBER', I4, ' DIAGNOSTICS EVERY', I3, ' STEPS' ,///)
INUMBER', I4,' AND',//,' GOING THROUGH Timestep NUMBER', I4,'.
DIAGNOSTICS EVERY', I3,' STEPS.' )

READ(5,100) XN, XN1, KONST
READ(5,100) RCLOUD, CONSTP, BDTYEM, DEN, CORMAS
READ(5,102) SIGMA, CIRP, AMP0, VMXIN, VMXOUT
READ(5,107) BOTBDY, TOPBDY, SIDBDY
ISYMA=IABS(ISYM)
IF(ISYMA.EQ.2.OR.ISYMA.EQ.3.OR.ISYMA.EQ.9)BOTBDY='WALL'
IF(BOTBDY.NE.'WALL'.AND.BOTBDY.NE.'FREE')BOTBDY='DIRICH'

FOR POLYTROPES, ISOADI MUST = 3.
C* IF(ISOADI.EQ.2.OR.ISOADI.EQ.3)WRITE(6,104)ITSTRT, ITSTOP
C* IF(ISOADI.EQ.1)WRITE(6,103)ITSTRT, ITSTOP, IDIAG
IF(VMXIN.LE.0.0) VMXIN=12.5
IF(VMXOUT.LE.0.0) VMXOUT=7.5
WRITE(6,110)
WRITE(6,100) XN, XN1, KONST
WRITE(6,100) RCLOUD, CONSTP, BDTYEM, DEN, CORMAS
WRITE(6,100) SIGMA, CIRP, AMP0, VMXIN, VMXOUT
WRITE(6,107) BOTBDY, TOPBDY, SIDBDY
C* IF(ISYMA.EQ.1)WRITE(6,112)
C* IF(ISYMA.EQ.2)WRITE(6,113)
C* IF(ISYMA.EQ.3)WRITE(6,114)
C* IF(ISYMA.EQ.8)WRITE(6,115)
C* IF(ISYMA.EQ.9)WRITE(6,116)
WRITE(6,117) BOTBDY, TOPBDY, SIDBDY
C* IF(ISOADI.EQ.3)WRITE(6,106) XN, XN1, KONST
INITIALIZE FOLLOWING PARAMETERS FOR ALL RUNS.

PI = 3.1415926535897E0
GRAV=667.32

IF(ISOADI.EQ.3)GRAV=1.0

DTHETA=2.0*PI/LMAX

IF(IABS(ISYM).GT.5)LMAX=1

IF(IABS(ISYM).EQ.3)DTHETA=0.5*DTHETA

GAMMA=XN1

CVHEAT=CURLYR/(XMU*(GAMMA-1.0))

THETA=-0.5*DTHETA
DO 4 L=1,LMAX
    THETA=THETA+DTHETA
    COSIGN(L) =COS(THETA)
    SIGN(L)=SIN(THETA)
    CONTINUE

TIME=0.0
DENEX=1.E-7*DEN
TMASS=0.0
ENEW=0.0
ELOST=0.0
EDIF=0.0
PHICHK=0.0
KLOCAT=0
MLOSTR=0.0
MLOSTZ=0.0
JLOSTZ=0.0
JLOSTR=0.0

THIS CALL MUST BE MADE TO INITIALIZE ARRAY WSAVE

FOR USE IN POT3. (WSAVE DIMENSIONED .GE. (2*N +15))

N=2*LMAX

IF(LMAX.GT.1)CALL RFFTI(N,WSAVE)

FINISHED SETTING UP GENERAL PARAMETERS.

SET UP R'S AND Z'S.
FIRST DETERMINE WHAT A1NEWR, A1NEWZ, ROF3N, AND ZOF3N ARE BY
EITHER READING IN A NEW MODEL STRUCTURE FROM INPUT FILE
TAPEN=FLOAT(NTAPE)
ICALL = 1

OR READING IN AN OLD MODEL FROM DISK...

IF(ITYPE.GT.1)GO TO 440

WRITE(6,105)NMODL,NTAPE

NM=NMODL-1

IF(NM.LE.0) GO TO 435
C* DO 433 J=1,MM
C* READ(NTAPE,2222)U,W,JN,RHO,ROF3N,ZOF3N,AINEW,AINEWZ,
C* IDELT,TIME,ELOST
C*433 CONTINUE
C*435 READ(NTAPE,2222)U,W,JN,RHO,ROF3N,ZOF3N,AINEW,AINEWZ,
C* IDELT,TIME,ELOST
READ(9)S,T, A,RHO,ROF3N,ZOF3N,AINEW,AINEWZ,DELT,TIME,MLOST,
1 MLOSTZ,JLOST,JLOSTZ
440 CONTINUE
C THEN CALCULATE R,Z,RHF, AND ZHF
DELR=R0F3N
DELZ=ZOF3N
R(3)=DELR
R(2)=0.0
R(1)=-R(3)
R(4)=2.0*DELR
DO 448 J=5,JMAX1
DELR=AINEW*DELR
R(J)=R(J-1)+DELR
448 CONTINUE
R(JMAX2) = R(JMAX1)+DELR
RCLOUD = R(JMAX1)
Z(3)=DELZ
Z(2)=0.0
Z(1)=-Z(3)
Z(4)=2.0*DELZ
DO 450 K=5,KMAX1
DELZ=AINEWZ*DELZ
Z(K)=Z(K-1)+DELZ
450 CONTINUE
Z(KMAX2) = Z(KMAX1) + DELZ
DO 452 J=1,JMAX1
452 RHF(J)=0.5*(R(J)+R(J+1))
DO 454 K=1,KMAX1
454 ZHF(K)=0.5*(Z(K)+Z(K+1))
IF(ITYPE.EQ.1) GO TO 550
C... FINISH SETTING UP INITIAL MODEL.
C SHIFT ANGULAR MOMENTUM TO HALF-INTEGER GRID POINTS.
ICALL = ITYPE + 1
IF(ISOADI.EQ.3) WRITE(6,'(A)')'ITYPE WRONG 2'
C SET UP RHO, U, W, AND JN, INCLUDING DESIRED PERTURBATION.
ICALL = ITYPE + 1
IF(ISOADI.EQ.3.AND.ITYPE.EQ.2) WRITE(6,'(A)')'ITYPE WRONG 3'
IF(ISOADI.EQ.3.AND.ITYPE.EQ.3) WRITE(6,'(A)')'ITYPE WRONG 4'
IF(ISOADI.EQ.3.AND.ITYPE.EQ.4) WRITE(6,'(A)')'ITYPE WRONG 5'
IF(ISOADI.EQ.3.AND.ITYPE.EQ.5) WRITE(6,'(A)')'ITYPE WRONG 6'
C AND SET UP RO,ZO,RHFO,ZHFO, AND G AND H.
C* DO 458 J=1,JMAX1
C* RO(J)=R(J)
C* RHFO(J)=RHF(J)
C* G(J)=0.0
C*458 CONTINUE
C* RO(JMAX2)=R(JMAX2)
C* G(JMAX2)=0.0

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
C* ZO(KMAX2)=Z(KMAX2)
C* H(KMAX2)=0.0
C* DO 460 K=1,KMAX1
C* ZO(K)=Z(K)
C* ZHFO(K)=ZHF(K)
C* H(K)=0.0
C*460 CONTINUE
C
C
C 550 CONTINUE
C FROM RHO, FIND P AND EPS FOR THE POLYTROPE.
DO 560 L=1,LMAX
DO 560 K=1,KMAX2
DO 560 J=1,JMAX2
C* CV(J,K,L)=CVHEAT
IF(ISOADI.NE.1)GO TO 554
C* P(J,K,L)=EPS(J,K,L)*(GAMMA-1.0)*RHO(J,K,L)
GO TO 560
554 CONTINUE
P(J,K,L)=KONST*RHO(J,K,L)**XN1
C* EPS(J,K,L) = P(J,K,L)/(RHO(J,K,L)*(GAMMA-1.0))
560 CONTINUE
C FROM S, T, A, AND RHO FIND U, W, AND VT
CALL VEL
RHFINV(2) = 1.0/RHF(2)**2
DO 565 J=3,JMAX1
JM=J-1
RD(J)=RHF(J)**2-RHF(JM)**2
RDM(J)=R(J)**2-RHF(JM)**2
RDP(J)=RHF(J)**2-R(J)**2
RINV(J)=2.0/(RHF(J)**2+RHF(JM)**2)
RHFINV(J) = 1.0/RHF(J)**2
565 CONTINUE
DO 570 K=2,KMAX1
KM=K-1
ZD(K)=ZHF(K)-ZHF(KM)
ZDM(K)=Z(K)-ZHF(KM)
ZDP(K)=ZHF(K)-Z(K)
570 CONTINUE
C FIND POTENTIALS FOR THE MODEL.
NZERO=0
NEIGHT=8
CALL SETBDY(NZERO,ISYM)
REDGE=0.0
CALL BDYGEN(MAXTRM,ISYM,REDGE)
CALL POT3(NEIGHT,NZERO,ISYM)
RETURN
END
SUBROUTINE GEM(ISYM,BOTBDY,TOPBDY,SIDBDY)
C FOR THE FULLY SECOND ORDER DIFFERENCE IN EVERYTHING.
PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2, 1
KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2
I.MAX64)
C COMMON /BLOK6/ DTHETA, COSIGN(LMAX), SSIGN(LMAX), PI, GRAV
C* COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX1), ZHF(KMAX1),
  1 COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX1), ZHF(KMAX1),
  1 ROF3N, ROF3N, A1NEWR, A1NEWZ
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX1), ZHF(KMAX1),
  1 RHFI(JMAX1), RHFI2(JMAX1), DR, DZ, DELTHI, RHFI3(JMAX1)
CHARACTER*6 BOTBDY, TOPBDY, SIDBDY
C IT IS ASSUMED THAT THE GRID IS UNIFORM IN Z AND R AND NOT MOVING
C OR Z. THERE SHOULD NEVER BE ANY REASON FOR
C NONUNIFORMITY IN THETA. IF Z IS NOT UNIFORM THE GEM ARRAYS WILL
C VARY IN K ALSO AND BE TWO DIMENSIONAL ARRAYS. ALSO IF THE G
C IN R OR Z IS ALLOWED TO MOVE THIS WILL HAVE TO BE UPDATED EVERY
C TIMESTEP. NOW IT IS CALLED ONCE AND FOR ALL.
  1 DELTHI=0.5/DTHETA
  2 DZ=Z(5)-Z(4)
  3 DR=R(5)-R(4)
  4 DZPI=DZ*PI/(2*LMAX)
  5 DZPI=DZ*DTHETA/2.0
  6 GEMS=DZ*DR
  7 GEMR=DZ*DTHETA
  8 DO 10 J=2, JMAX1
  9   JP=J+1
 10  R2DIF=R(JP)**2-R(J)**2
 11   VOLMI(J)=1.0/(DZPI*R2DIF)
 12   GEMZ(J)=DTHETA/2.0*R2DIF
 13   DT(J)=RHF(J)*DTHETA
 14   RHFI3(J)=1.0/(RHF(J)**3)
 15   CONTINUE
 16   RHFI2(J)=1.0/RHF(J)**2
 17   DO 501 J=2, JMAX1
 18     RHFI(J)=1.0/RHF(J)
 19     RHFI2(J)=RHFI(J)/RHF(J)
 20  501 CONTINUE
C GEMR ARE MISSING AN R(J)* FACTOR EACH
C SINCE THIS IS A FUNCTION OF VELOCITY IT CAN NOT BE COMPUTED JUST
C ONCE.
C CALL SETUP2
RETURN
END
SUBROUTINE SETUP2
C
PARAMETER (JMAX=32, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
  1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
  2 LMAX=64)
C C* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
REAL JN
COMMON /EOM/ S(JMAX2, KMAX2, LMAX), T(JMAX2, KMAX2, LMAX),
  1 A(JMAX2, KMAX2, LMAX), U(JMAX2, KMAX2, LMAX),
  2 W(JMAX2, KMAX2, LMAX), JN(JMAX2, KMAX2, LMAX),
  3 OMEGA(JMAX2, KMAX2, LMAX), VT(JMAX2, KMAX2, LMAX)
COMMON /POIS/ PHI(JMAX2, KMAX2, LMAX), RHO(JMAX2, KMAX2, LMAX)
C A AND RHO ARE ZEROED HERE ONCE SO NOTHING WILL BE IN THESE ZONES
C HOPEFULLY IT WILL STAY THAT WAY WITH A COMPLETELY ABSORBING EDGE
C BUT MLOSTZ AND MLOSTR WILL KEEP TRACK OF MASS LOST IN EOC AND
C JLOSTZ AND JLOSTR WILL KEEP TRACK OF ANGULAR MOMENTUM LOST IN EO

DMAX=RHO(JMAX2,KMAX2,1)
DO 2 L=1,LMAX
DO 1 K=1,KMAX1
A(JMAX1,K,L)=0.0
A(JMAX2,K,L)=0.0
S(JMAX1,K,L)=0.0
S(JMAX2,K,L)=0.0
T(JMAX1,K,L)=0.0
T(JMAX2,K,L)=0.0
RHO(JMAX1,K,L)=DMAX*1.0E-07
RHO(JMAX2,K,L)=DMAX*1.0E-07
1 CONTINUE
2 CONTINUE
DO 4 L=1,LMAX
DO 3 J=2,JMAX
A(J,KMAX1,L)=0.0
A(J,KMAX2,L)=0.0
S(J,KMAX1,L)=0.0
S(J,KMAX2,L)=0.0
T(J,KMAX1,L)=0.0
T(J,KMAX2,L)=0.0
RHO(J,KMAX1,L)=DMAX*1.0E-07
RHO(J,KMAX2,L)=DMAX*1.0E-07
3 CONTINUE
4 CONTINUE
RETURN
END
SUBROUTINE SOST(ISYM,BOTBDY,TOPBDY,SIDBDY)

C FULLY SECOND ORDER VERSION.
C EVERYTHING RHO, S, T, AND A ARE ALL CENTERED AT THE SAME CELL

PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)

COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SSIGN(LMAX), PI,GRAV
COMMON /BLOK7/ RCLOUD,CONSTP,DELT,BDYTEM,DEN,TIME,CORMAS
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX1),
1 ROF3N,ZOF3N,A1NEWR,A1NEWZ
REAL JN

* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /STATES/ P(JMAX2,KMAX2,LMAX)
COMMON /GEX/ GEMS,GEMR,GEMZ(JMAX1),VOLMi(JMAX1),DT(JMAX1),
1 RHFi(JMAX1),RHFi2(JMAX1),DR,DZ,DELTHi,RHFi3(JMAX1)
CHARACTER*6 BOTBDY,TOPBDY,SIDBDY

C-------------------CONDITIONAL VECTOR MERGE-------------------
LOGICAL ILL,ILL2

C ON THE FPS-264 AFIX IS WRITTEN AS:
C AFIX(Ill)=FLOAT(AND(SHIFT(Ill,-31),1))
C AN ALTERNATE WAY TO WRITE AFIX ON THE FPS-264 IS:
C AFIX(Ill)=FLOAT(SHIFT(Ill,-63))
C THIS SAVES SOME TIME BUT ENGINEERING IS CHECKING OUT IF THIS
C WILL ALWAYS WORK. FOR LOGICAL .TRUE. THE FPS PUTS A 1 IN BIT 32
C 11. FOR .FALSE. THE FPS PUT A 0 IN BIT 32 AND 11. THE OTHER BI
C IN A LOGICAL ARE IGNORED.
C
C ON A VAX USING VMS 4.5 AFIX CAN BE WRITTEN AS:
C AFIX(Ill)=ABS(FLOAT(Ill))
C THE VAX MAKES .TRUE. -1 AND .FALSE. 0.
C
C ON AN IBM AFIX MUST BE MADE A FUNCTION BECAUSE OF THE SILLY WAY
C THE IBM COMPILER CHECKS VARIABLE Typing AT Compile TIME.
FUNCTION AFIX(III)
AFIX=FLOAT(III)
RETURN
END

C ON THE IBM OF COURSE THE FUNCTION AFIX(III) CAN NOT BE PLACED HE
C AT LEAST THAT IS THE WAY VSFORT WAS ON 3-9-87.
C
C ON ANY CRAY MACHINE CVMGm, CVmGp, CVMGz, CVMGn, AND CVMGT ARE AL
C DEFINED WITHIN THE FORTRAN LANGUAGE AS EXTENSIONS.
C
C-------------------CONDITIONAL VECTOR MERGE FUNCTIONS DEFINED-------------------
CVMGM(F,B,C)=AFIX(C .LT. 0.0)*F+AFIX(C .GE. 0.0)*B
CVMGP(F,B,C)=AFIX(C .GE. 0.0)*F+AFIX(C .LT. 0.0)*B
CVMGP1(F,B,C)=AFIX(C .GT. 0.0)*F+AFIX(C .LT. 0.0)*B
CVMG2(F,B,C)=AFIX(C .EQ. 0.0)*F+AFIX(C .NE. 0.0)*B
CVMGN(F,B,C)=AFIX(C .NE. 0.0)*F+AFIX(C .EQ. 0.0)*B
CVMGT(F,B,ILL)=AFIX(ILL)*F+AFIX(.NOT. ILL)*B
CVMGT2(F,B,ILL2)=AFIX(ILL2)*F+AFIX(.NOT. ILL2)*B
CVMZPI(B)=CVMGT(0.0,1.0,B .LE. 1.0E-100)/
   CVMGT2(1.0,B,B .LE. 1.0E-100)

ISYMA=IABS(ISYM)
DRI2=0.5/DR
DZI2=0.5/DZ

C------------------------------------------------------------------
C--------COMPUTE THE NEW S'S FROM THE SOURCES---------------------
C******* FOR THE L=LMAX CASE THE NEW S'S************************
L=LMAX
LP=1
LM=LMAX-1
DO 728 K=2,KMAX
   KP=K+1
   DO 727 J=2,JMAX
      JP=J+1
      JM=J-1
   1 P(JM,K,L)-P(JP,K,L))*DRI2+ 
   2 A(J,K,L)*A(J,K,L)*RHFI3(J)*CVMZPI(RH0(J,K,L)))*DELT
727 CONTINUE
728 CONTINUE
C******* FOR THE L=1 CASE THE NEW S'S*****************************
L=1
LP=2
LM=LMAX
DO 730 K=2,KMAX
   KP=K+1
   DO 729 J=2,JMAX
      JP=J+1
      JM=J-1
   1 P(JM,K,L)-P(JP,K,L))*DRI2+ 
   2 A(J,K,L)*A(J,K,L)*RHFI3(J)*CVMZPI(RH0(J,K,L)))*DELT
729 CONTINUE
730 CONTINUE
C******* FOR THE L=2,LMAX-1 CASE THE NEW S'S********************
DO 733 L=2,LMAX-1
   LP=L+1
   LM=L-1
   DO 732 K=2,KMAX
      KP=K+1
      DO 731 J=2,JMAX
         JP=J+1
         JM=J-1
   1 P(JM,K,L)-P(JP,K,L))*DRI2+ 
   2 A(J,K,L)*A(J,K,L)*RHFI3(J)*CVMZPI(RH0(J,K,L)))*DELT
731 CONTINUE
732 CONTINUE
733 CONTINUE
C------------------------------------------------------------------
C-----------------NOW DO T----------------------------------------
C--------COMPUTE THE NEW T'S FROM THE FLUX'S AND SOURCES --------
C****** FOR THE L=LMAX CASE THE NEW T'S************************
L=LMAX
LP=1
DO 828 K=2,KMAX
    KP=K+1
    KM=K-1
    DO 827 J=2,JMAX
        JP=J+1
        T(J,K,L)=T(J,K,L)+(RHO(J,K,L)*(PHI(J,KM,L)-PHI(J,KP,L))+
         1 P(J,KM,L)-P(J,KP,L))*DZI2*DELT
    827 CONTINUE
828 CONTINUE
C****** FOR THE L=1,LMAX-1 CASE THE NEW T'S********************
DO 833 L=1,LMAX-1
    LP=L+1
    DO 832 K=2,KMAX
        KP=K+1
        KM=K-1
        DO 831 J=2,JMAX
            JP=J+1
            T(J,K,L)=T(J,K,L)+(RHO(J,K,L)*(PHI(J,KM,L)-PHI(J,KP,L))+
             1 P(J,KM,L)-P(J,KP,L))*DZI2*DELT
        831 CONTINUE
832 CONTINUE
833 CONTINUE
CCCCcccccccccccccccccccccCccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
NEUMANN CONDITION IF SYMMETRY THRU EQUATORIAL PLANE ASS

DO 295 L=1,LMAX
DO 295 J=1,JMAX1
S(J,1,L) = S(J,2,L)
T(J,1,L) = -T(J,2,L)

CONTINUE

GO TO 910

CONTINUE

C IF BOTBDY = 'WALL' OR 'FREE', THEN REFLECTION SYMMETRY
C AMOUNTS TO THE SAME THING AS NEUMANN CONDITION.
IF(BOTBDY.EQ.'WALL'.OR.BOTBDY.EQ.'FREE') GO TO 294

C IF BOTBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
C IS ASSUMED AND NO MODIFICATION OF K=1 IS MADE.

910 CONTINUE

C RETURN
END

SUBROUTINE HYDROS(ISYM,BOTBDY,TOPBDY,SIDBDY)

FULLY SECOND ORDER VERSION.
EVERYTHING RHO, S, T, AND A ARE ALL CENTERED AT THE SAME CELL

PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)

COMMON /BLOK6/ DTHETA, COSIGN(LMAX), SSIGN(LMAX), PI, GRAV
COMMON /BLOK7/ RCL, CONSP, DELT, BDTEM, DEN, TIME, CORRAS
COMMON /GRID/ R(JMAX2), Z(KMAX2), RHF(JMAX2), ZOF3N, ALNEWZ
REAL JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX), T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX), U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX), JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX), VT(JMAX2,KMAX2,LMAX)

COMMON /POIS/ PH(JMAX2,KMAX2,LMAX), RHO(JMAX2,KMAX2,LMAX)
COMMON /STATES/ P (JMAX2, KMAX2, LMAX)
COMMON /GEX/ GEMR, GEMZ(JMAX1), VOLMI(JMAX1), DT(JMAX1),
1 RHF1(JMAX1), RHF2(JMAX1), DR, DZ, DELTHI, RHF3(JMAX1)
COMMON /FLUX/ FR(JMAX1,KMAX1,LMAX), FZ(JMAX1,KMAX1,LMAX),
1 FT(JMAX1,KMAX1,LMAX), UTIL(JMAX1,KMAX1,LMAX)
CHARACTER*6 BOTBDY, TOPBDY, SIDBDY

-------------CONDITIONAL VECTOR MERGE-----------------------
LOGICAL ILL, ILL2

ON THE FPS-264 AFIX IS WRITTEN AS:
AFIX(ILL)=FLOAT(AND(SHIFT(ILL,-31),1))
AN ALTERNATE WAY TO WRITE AFIX ON THE FPS-264 IS:
AFIX(ILL)=FLOAT(SHIFT(ILL,-63))
THIS SAVES SOME TIME BUT ENGINEERING IS CHECKING OUT IF THIS
WILL ALWAYS WORK. FOR LOGICAL .TRUE. THE FPS PUTS A 1 IN BIT 32
11. FOR .FALSE. THE FPS PUT A 0 IN BIT 32 AND 11. THE OTHER BI
C IN A LOGICAL ARE IGNORED.
C ON A VAX USING VMS 4.5 AFIX CAN BE WRITTEN AS:
C
C       AFIX(ILL) = ABS(FLOAT(ILL))
C
C THE VAX MAKES .TRUE. -1 AND .FALSE. 0.
C
C ON AN IBM AFIX MUST BE MADE A FUNCTION BECAUSE OF THE SILLY WAY
C THE IBM COMPILER CHECKS VARIABLE TYPING AT COMPILATE TIME.
C
FUNCTION AFIX(III)
C
AFIX = FLOAT(III)
RETURN
END
C
C ON THE IBM OF COURSE THE FUNCTION AFIX(III) CAN NOT BE PLACED HE
C AT LEAST THAT IS THE WAY VSFORT WAS ON 3-9-87.
C
C ON ANY CRAY MACHINE CVMGM, CVMGP, CVMGZ, CVMGN, AND CVMGT ARE AL
C DEFINED WITHIN THE FORTRAN LANGUAGE AS EXTENSIONS.
C
C------CONDITIONAL VECTOR MERGE FUNCTIONS DEFINED.------
C
CVMGM(F,B,C) = AFIX(C .LT. 0.0)*F + AFIX(C .GE. 0.0)*B
CVMGP(F,B,C) = AFIX(C .GE. 0.0)*F + AFIX(C .LT. 0.0)*B
CVMG1(F,B,C) = AFIX(C .GT. 0.0)*F + AFIX(C .LE. 0.0)*B
CVMGZ(F,B,C) = AFIX(C .EQ. 0.0)*F + AFIX(C .NE. 0.0)*B
CVMGN(F,B,C) = AFIX(C .NE. 0.0)*F + AFIX(C .EQ. 0.0)*B
CVMGT(F,B,ILL) = AFIX(ILL)*F + AFIX(.NOT. ILL)*B
CVMGTP(F,B,ILL2) = AFIX(ILL2)*F + AFIX(.NOT. ILL2)*B
CVMZPI(B) = CVMGT(0.0,1.0,B .LE. 1.0E-100)/
1 CVMGTP(1.0,B,B .LE. 1.0E-100)
C
C------VAN LEER SLOPES FOR S AND T------
C
SLOPEA(F,B,C) = CVMGT(0.,1.,ABS(C-F) .LE. 1.0E-100
1 .OR. (C-B)*(B-F) .LE. 0. )
2 *( 2.*(B-C)*(F-B) )/
3 CVMGTP(1.,F-C,
4 ABS(C-F) .LE. 1.0E-100 .OR. (C-B)*(B-F) .LE. 0.0)
C
C------FIRST ORDER SLOPES FOR S AND T------
C
SLOPEA(F,B,C)=0.0
C
C------VAN LEER INTERPOLATION FOR S AND T------
C
C IN THE J ,K OR L DIRECTION WHERE DX DOES NOT CHANGE WITH GRID CE
C
VLI(Q,Q1,DX,V,SLOPE,SLOPE1)=
1 CVMGP(Q1+(DX-V*DELT)*SLOPE1/(2.*DX),
2 Q -(DX+V*DELT)*SLOPE1/(2.*DX),V)
C
C------FIRST ORDER INTERPOLATION FOR S AND T------
C
C IN THE J ,K OR L DIRECTION WHERE DX DOES NOT CHANGE WITH GRID CE
C
VLI(Q,Q1,DX,V,SLOPE,SLOPE1)=CVMGP(Q1,Q,V)
C
C
ISYMA=IABS(ISYM)
HALFDT = 0.5*DELT
DRI2=0.5/DR
DZI2=0.5/DZ
C------VAN LEER SLOPES FOR S IN THE J DIRECTION------
DO 703 L=1,LMAX

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 702 K=2,KMAX
DO 701 J=2,JMAX
JP=J+1
JM=J-1
UTIL(J,K,L)=SLOPEA(S(JP,K,L),S(J,K,L),S(JM,K,L))
701 CONTINUE
702 CONTINUE
703 CONTINUE

C------------------------FLUX'S FOR S IN THE J DIRECTION------------------------
DO 706 L=1,LMAX
DO 705 K=2,KMAX
FR(2,K,L)=0.0
DO 704 J=3,JMAX
JM=J-1
TEMP=U(J,K,L)
C-----------FLUX'S FOR S IN THE J DIRECTION USING R*(J)-----------
FR(J,K,L)=VLI(S(J,K,L),S(JM,K,L),DR,TEMP,1)*UTIL(J,K,L)*UTIL(JM,K,L)*TEMP*GEMR*(R(J)-TEMP*HALFDT)
C----------FLUX'S FOR S IN THE J DIRECTION WITHOUT USING R*(J)----------
C FR(J,K,L)=VLI(S(J,K,L),S(JM,K,L),DR,TEMP,1)*TEMP*GEMR*R(J)
C------------------------------------------------------------------
704 CONTINUE
705 CONTINUE
706 CONTINUE

C------------------------------------------------------------------
C----------VAN LEER SLOPES FOR S IN THE K DIRECTION ------------
DO 709 L=1,LMAX
DO 708 K=2,KMAX
KP=K+1
KM=K-1
DO 707 J=2,JMAX
UTIL(J,K,L)=SLOPEA(S(J,KP,L),S(J,K,L),S(J,KM,L))
707 CONTINUE
708 CONTINUE
709 CONTINUE

C------------------------FLUX'S FOR S IN THE K DIRECTION------------------------
DO 712 L=1,LMAX
DO 711 K=2,KMAX
KM=K-1
DO 710 J=2,JMAX
TEMP=W(J,K,L)
FZ(J,K,L)=VLI(S(J,K,L),S(J,KM,L),DZ,TEMP,1)*UTIL(J,K,L)*UTIL(J,KM,L)*TEMP*GEMZ(J)
710 CONTINUE
711 CONTINUE
712 CONTINUE

C------------------------VAN LEER SLOPES FOR S IN THE L DIRECTION------------------------
C** FOR THE L=1 CASE SLOPES **********************************************************
L=1
LP=2
LM=LMAX
DO 714 K=2,KMAX
DO 713 J=2,JMAX
UTIL(J,K,L)=SLOPEA(S(J,K,LP),S(J,K,L),S(J,K,LM))
713 CONTINUE
714 CONTINUE
C******** FOR THE L=LMAX CASE SLOPES *******************************
   L=LMAX
   LP=1
   LM=LMAX-1
   DO 716 K=2,KMAX
   DO 715 J=2,JMAX
   UTIL(J,K,L)=SLOPEA(S(J,K,LP),S(J,K,L),S(J,K,LM))
715 CONTINUE
716 CONTINUE
C******** FOR THE L=2,LMAX-1 CASE SLOPES *****************************
   DO 719 L=2,LMAX-1
   LP=L+1
   LM=L-1
   DO 718 K=2,KMAX
   DO 717 J=2,JMAX
   UTIL(J,K,L)=SLOPEA(S(J,K,LP),S(J,K,L),S(J,K,LM))
717 CONTINUE
718 CONTINUE
719 CONTINUE
C------------------------------------------------------------------
C--------------FLUX' S FOR S IN THE L DIRECTION -------------------
Q******* FOR THE L=1 CASE FLUX' S ****************************************
   L=1
   LM=LMAX
   DO 721 K=2,KMAX
   DO 720 J=2,JMAX
   TEMP=VT(J,K,L)
   FT(J,K,L)=VLI(S(J,K,L),S(J,K,LM),DT(J),TEMP,
   1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
720 CONTINUE
721 CONTINUE
C******** FOR THE L=2,LMAX CASE FLUX' S ******************************
   DO 726 L=2,LMAX
   LM=L-1
   DO 725 K=2,KMAX
   DO 724 J=2,JMAX
   TEMP=VT(J,K,L)
   FT(J,K,L)=VLI(S(J,K,L),S(J,K,LM),DT(J),TEMP,
   1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
724 CONTINUE
725 CONTINUE
726 CONTINUE
C------------------------------------------------------------------
C--------COMPUTE THE NEW S'S FROM THE FLUX' S----------------------
C******* FOR THE L=LMAX CASE THE NEW S'S*****************************
   L=LMAX
   LP=1
   LM=LMAX-1
   DO 728 K=2,KMAX
KP=K+1
DO 727 J=2,JMAX
JP=J+1
JM=J-1
1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
727 CONTINUE
728 CONTINUE
c******* FOR THE L=1 CASE THE NEW S’S***********************
L=1
LP=2
LM=LMAX
DO 730 K=2,KMAX
KP=K+1
DO 729 J=2,JMAX
JP=J+1
JM=J-1
1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
729 CONTINUE
730 CONTINUE
c******* FOR THE L=2,LMAX-1 CASE THE NEW S’S***********************
DO 733 L=2,LMAX-1
LP=L+1
LM=L-1
DO 732 K=2,KMAX
KP=K+1
DO 731 J=2,JMAX
JP=J+1
JM=J-1
1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
731 CONTINUE
732 CONTINUE
733 CONTINUE
c------------------------------------------------------------------
c-----------------NOW DO T-----------------------------------------
c-------------VAN LEER SLOPES FOR T IN THE J DIRECTION-------------
DO 803 L=1,LMAX
DO 802 K=2,KMAX
DO 801 J=2,JMAX1
JP=J+1
JM=J-1
UTIL(J,K,L)=SLOPEA(T(JP,K,L),T(J,K,L),T(JM,K,L))
801 CONTINUE
802 CONTINUE
803 CONTINUE
c------------------------------------------------------------------
c-------------FLUX’S FOR T IN THE J DIRECTION---------------------
DO 806 L=1,LMAX
DO 805 K=2,KMAX
FR(2,K,L)=0.0
DO 804 J=3,JMAX1
JM=J-1
804 CONTINUE
805 CONTINUE
806 CONTINUE
\[ \text{TEMP} = U(J,K,L) \]

---FLUX'S FOR T IN THE J DIRECTION USING \( R(J) \)---

\[ FR(J,K,L) = \text{VLI}(T(J,K,L), T(JM,K,L), DR, \text{TEMP}, \]

\[ 1 \text{ UTIL}(J,K,L), UTIL(JM,K,L)) \times \text{TEMP} \times \text{GEMR}(R(J) - \text{TEMP} \times \text{HALFDT}) \]

---FLUX'S FOR T IN THE J DIRECTION WITHOUT USING \( R(J) \)---

\[ C \]

\[ \text{FR}(J,K,L) = \text{VLI}(T(J,K,L), T(JM,K,L), DR, \text{TEMP}, \]

\[ 1 \text{ UTIL}(J,K,L), UTIL(JM,K,L)) \times \text{TEMP} \times \text{GEMR}(R(J)) \]

C------------------------------------------------------------------

804 CONTINUE
805 CONTINUE
806 CONTINUE

C------------------------------------------------------------------

C----------VAN LEER SLOPES FOR T IN THE K DIRECTION ------------

DO 809 L=1,LMAX
DO 808 K=2,KMAX
KP=K+1
KM=K-1
DO 807 J=2,JMAX
\[ \text{UTIL}(J,K,L) = \text{SLOPEA}(T(J,KP,L), T(J,K,L), T(J,KM,L)) \]

807 CONTINUE
808 CONTINUE
809 CONTINUE

C------------------------------------------------------------------

C--------------FLUX'S FOR T IN THE K DIRECTION ------------------

DO 812 L=1,LMAX
DO 811 K=2,KMAX
KM=K-1

DO 810 J=2,JMAX
\[ \text{TEMP} = W(J,K,L) \]
\[ FZ(J,K,L) = \text{VLI}(T(J,K,L), T(J,KM,L), DZ, \text{TEMP}, \]

\[ 1 \text{ UTIL}(J,K,L), UTIL(J,KM,L)) \times \text{TEMP} \times \text{GEMZ}(J) \]

810 CONTINUE
811 CONTINUE
812 CONTINUE

C------------------------------------------------------------------

C----------VAN LEER SLOPES FOR T IN THE L DIRECTION -------------

C*** FOR THE L=1 CASE SLOPES ***********************************

\[ L = 1 \]
\[ LP = 2 \]
\[ LM=\text{LMAX} \]

DO 814 K=2,KMAX
DO 813 J=2,JMAX
\[ \text{UTIL}(J,K,L) = \text{SLOPEA}(T(J,K,LP), T(J,K,L), T(J,K,LM)) \]

813 CONTINUE
814 CONTINUE

C*** FOR THE L=LMAX CASE SLOPES ***********************************

\[ L = \text{LMAX} \]
\[ LP = 1 \]
\[ LM=\text{LMAX}-1 \]

DO 816 K=2,KMAX
DO 815 J=2,JMAX
\[ \text{UTIL}(J,K,L) = \text{SLOPEA}(T(J,K,LP), T(J,K,L), T(J,K,LM)) \]

815 CONTINUE
816 CONTINUE

C*** FOR THE L=2,LMAX-1 CASE SLOPES ***********************************

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 819 L=2,LMAX-1
   LP=L+1
   LM=L-1
   DO 818 K=2,KMAX
      DO 817 J=2,JMAX
         UTIL(J,K,L)=SLOPEA(T(J,K,LP),T(J,K,L),T(J,K,LM))
      817 CONTINUE
   818 CONTINUE
   819 CONTINUE

C------------------------------------------------------------------
C--------------FLUX'S FOR T IN THE L DIRECTION------------------
C******** FOR THE L=1 CASE FLUX'S *********************************
   L=1
   IM=LMAX
   DO 821 K=2,KMAX
      DO 820 J=2,JMAX
         TEMP=VT(J,K,L)
         FT(J,K,L)=VLI(T(J,K,L),T(J,K,IM),DT(J),TEMP,
            1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
      820 CONTINUE
   821 CONTINUE

C******** FOR THE L=2,LMAX CASE FLUX'S *******************************
   DO 826 L=2,LMAX
      IM=L-1
      DO 825 K=2,KMAX
         DO 824 J=2,JMAX
            TEMP=VT(J,K,L)
            FT(J,K,L)=VLI(T(J,K,L),T(J,K,LM),DT(J),TEMP,
               1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
         824 CONTINUE
      825 CONTINUE
   826 CONTINUE

C------------------------------------------------------------------
C--------COMPUTE THE NEW T'S FROM THE FLUX'S----------------------
C******** FOR THE L=LMAX CASE THE NEW T'S****************************
   L=LMAX
   LP=1
   DO 828 K=2,KMAX
      KP=K+1
      KM=K-1
      DO 827 J=2,JMAX
         JP=J+1
            1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
      827 CONTINUE
   828 CONTINUE

C******** FOR THE L=1,LMAX-1 CASE THE NEW T'S************************
   DO 833 L=1,LMAX-1
      LP=L+1
      DO 832 K=2,KMAX
         KP=K+1
         KM=K-1
         DO 831 J=2,JMAX
            JP=J+1
               Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\[ T(J,K,L) = T(J,K,L) - (FR(JP,K,L) - FR(J,K,L) + FZ(J,KP,L) - FZ(J,K,L) - FT(J,K,LP) - FT(J,K,L) \times VOLMI(J) \times DELT \]

CONTINUE
CONTINUE
CONTINUE

FINISHED MAJOR WORK ... NOW CLEAN UP

SET BOUNDARY CONDITIONS ON \( S \) AND \( T \)

Z-AXIS.
NEUMANN CONDITION HOLDS UNLESS \( ISYMA = 1 \) OR \( 2 \).

\[ LHAF = LMAX/2 \]
\[ LSTOP = LHAF \]
\[ IF (ISYMA.NE.1 .AND. ISYMA.NE.2) LSTOP = LMAX \]
DO 292 \( L = 1, LSTOP \)
\[ LP = L + LHAF \]
\[ IF (LSTOP.EQ.LMAX) LP = L \]
DO 292 \( K = 2, KMAX \)
\[ S(1,K,L) = -S(2,K,LP) \]
\[ S(1,K,LP) = -S(2,K,L) \]
\[ T(1,K,L) = T(2,K,LP) \]
\[ T(1,K,LP) = T(2,K,L) \]

CONTINUE

BOTTOM OF GRID.
IF \( ISYMA.EQ.1 .OR. ISYMA.EQ.8 \) GO TO 296

NEUMANN CONDITION IF SYMMETRY THRU EQUATORIAL PLANE ASS

DO 294 \( L = 1, LMAX \)
DO 294 \( J = 1, JMAX1 \)
\[ S(J,1,L) = S(J,2,L) \]
\[ T(J,1,L) = -T(J,2,L) \]

CONTINUE
GO TO 910

CONTINUE

IF \( BOTBDY = 'WALL' \) OR \( 'FREE' \), THEN REFLECTION SYMMETRY
AMOUNTS TO THE SAME THING AS NEUMANN CONDITION.
IF \( BOTBDY.EQ.\ 'WALL' .\ OR. \ BOTBDY.EQ.\ 'FREE' \) GO TO 294
IF \( BOTBDY.NE. \ 'FREE' \) OR \( WALL \), THEN A DIRICHLET BOUNDARY C
IS ASSUMED AND NO MODIFICATION OF \( K=1 \) IS MADE.

CONTINUE
RETURN
END

SUBROUTINE SOA(ISYM,BOTBDY,TOPOBDY,SIDBDY)
FOR THE FULLY SECOND ORDER DIFFERENCE WITH EVERYTHING \( \rho \), \( A \), \( S \), CENTERED IN THE SAME PLACE AT THE CENTER OF A CELL.
PARAMETER (JMAX=64, JMAX1=JMAX+1, JMAX2=JMAX+2, KMAX=16, KMAX1=KMAX+1, KMAX2=KMAX+2, LMAX=64)
C
COMMON /BLOK6/ DTHETA, COSIGN(1MAX), SIGN(1MAX), PI, GRAV
COMMON /BLOK7/ RCL, CONSTP, DELT, BDM, NAM, TIME, CORMAS
COMMON /GRID/ R(1MAX2), Z(1MAX2), RHF(1MAX2), ZHF(1MAX1),
1 ROF3N, ZOF3N, AINWR, ALNEWZ
REAL JN
C* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
C* EXCEPT RIGHT NOW JN IS USED IN DELTA TO CALCULATE THE TIMESTEP
COMMON /EOM/ S(JMAX2,1MAX2,1MAX), T(JMAX2,1MAX2,1MAX),
1 A(JMAX2,1MAX2,1MAX), U(JMAX2,1MAX2,1MAX),
2 W(JMAX2,1MAX2,1MAX), JN(JMAX2,1MAX2,1MAX),
3 OMEGA(JMAX2,1MAX2,1MAX), VT(JMAX2,1MAX2,1MAX)
COMMON /POIS/ PHI(JMAX2,1MAX2,1MAX), RHO(JMAX2,1MAX2,1MAX)
COMMON /STATES/ P(JMAX2,1MAX2,1MAX)
COMMON /NORMAL/ CIRP
COMMON /FREEZ/ DENEX
REAL JLOSTR, JLOSTZ, MLOSTR, MLOSTZ
COMMON /ANGMOM/ JLOSTR, JLOSTZ, MLOSTR, MLOSTZ
COMMON /GEX/ GEMS, GEMR, GEMZ(1MAX1), VOLM(1MAX1), DT(1MAX1),
1 RHFI(JMAX1), RHFI2(JMAX1), DR, DZ, DELTHI, RHFI3(JMAX1)
CHARACTER*6 BOTBDY, TOPBDY, SIBDY
ISYMA=IABS(ISYM)
C------------------------------------------------------------------
C--------COMPUTE THE NEW A'S FROM THE SOURCES---------------------
C******* FOR THE L=LMAX CASE THE NEW A'S************************
L=LMAX
LP=1
LM=LMAX-1
DO 728 K=2,1MAX
KP=K+1
DO 727 J=2,1MAX
JP=J+1
1 P(J,K,LP)-P(J,K,LM))*DELTHI
727 CONTINUE
728 CONTINUE
C******* FOR THE L=1 CASE THE NEW A'S************************
L=1
LP=2
LM=1MAX
DO 730 K=2,1MAX
KP=K+1
DO 729 J=2,1MAX
JP=J+1
1 P(J,K,LP)-P(J,K,LM))*DELTHI
729 CONTINUE
730 CONTINUE
C******* FOR THE L=2,1MAX-1 CASE THE NEW A'S********************
DO 733 L=2,1MAX-1
LP=L+1
LM=L-1
DO 732 K=2,1MAX
KP=K+1

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

DO 731 J=2,JMAX
JP=J+1
P(J,K,LP)-P(J,K,LM))*DELTHI
731 CONTINUE
732 CONTINUE
733 CONTINUE

CC... SMOOTH A(J,K,L) OVER J=2 AND 3 TO PRODUCE UNIFORM ROTATION
OVER THESE ZONES.
RR2=RHF(2)**2
RR3=RHF(3)**2
ARSMO2=R(3)*R(3)
ARSMO3=R(4)*R(4) - ARSMO2
ARRAT=ARSMO3/ARSMO2
DO 300 K=2,KMAX1
R2=0.0
XR3=0.0
A2=0.0
A3=0.0
DO 301 L=1,LMAX
R2=R2 + RH0(2,K,L)
XR3=XR3 + RH0(3,K,L)
A2=A2 + A(2,K,L)
A3=A3 + A(3,K,L)
301 CONTINUE
RSQM=R2*RR2 + ARRAT*XR3*RR3
ATOT=A2 + ARRAT*A3
OMUR=ATOT/RSQM
DO 302 L=1,LMAX
A(2,K,L)=RR2*OMUR*RHO(2,K,L)
A(3,K,L)=RR3*OMUR*RHO(3,K,L)
302 CONTINUE

C... REDEFINE OMEGA AND JN.
DO 500 L=1,LMAX
DO 500 K=2,KMAX
DO 500 J=2,JMAX1
JN(J,K,L) = A(J,K,L)/RHO(J,K,L)
C* OMEGA(J,K,L) = JN(J,K,L) * RHFINV(J)
OMEGA(J,K,L) = JN(J,K,L) * RHFI2(J)
500 CONTINUE

C... TO PREVENT TIME STEP FROM BEING CONTROLLED BY HIGH ANGULAR

C VELOCITIES IN LOW DENSITY REGIONS, SET JN AND OMEGA TO ZERO
C IF RHO.LE.D AND IF OMEGA.GT.FUDGE*(CENTRAL OMEGA).
D=1.0E3*DENEX
FUDGE = 0.25
ROTMAX = 2.0*PI/CIRP*FUDGE
DO 600 L=1,LMAX

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 600 K=2,KMAX
DO 600 J=2,JMAX
IF(RHO(J,K,L).GT.D)GO TO 600
IF(ABS(OMEGA(J,K,L)).LE.ROTMAX)GO TO 600
JN(J,K,L)=0.
OMEGA(J,K,L)=0.0
A(J,K,L)=0.0
VT(J,K,L)=0.0
600 CONTINUE

C
C
C
C...  SET BOUNDARY CONDITIONS ON RHO, A, JN, AND OMEGA.
C
C...  Z-AXIS.
C
NEUMANN CONDITION HOLDS UNLESS ISYMA = 1 OR 2.
LHAF=IMAX/2
LSTOP=LHAF
IF(ISYMA.NE.1.AND.ISYMA.NE.2)LSTOP=IMAX
DO 292 L=1,LSTOP
LP=L+LHAF
IF(LSTOP.EQ.IMAX)LP=L
DO 292 K=2,KMAX
A(1,K,L) = A(2,K,LP)
A(1,K,LP) = A(2,K,L)
JN(1,K,L) = JN(2,K,LP)
JN(1,K,LP) = JN(2,K,L)
OMEGA(1,K,L) = OMEGA(2,K,LP)
OMEGA(1,K,LP) = OMEGA(2,K,L)
292 CONTINUE
C...  BOTTOM OF GRID.
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)GO TO 296
C
NEUMANN CONDITION IF SYMMETRY THRU EQUATORIAL PLANE ASS
DO 295 J=1,JMAX
A(J,1,L) = A(J,2,L)
JN(J,1,L) = JN(J,2,L)
OMEGA(J,1,L) = OMEGA(J,2,L)
295 CONTINUE
GO TO 910
296 CONTINUE
C
IF BOTBDY = 'WALL' OR 'FREE', THEN REFLECTION SYMMETRY
AMOUNTS TO THE SAME THING AS NEUMANN CONDITION.
IF(BOTBDY.EQ.'WALL'.OR.BOTBDY.EQ.'FREE')GO TO 294
C
IF BOTBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
IS ASSUMED AND NO MODIFICATION OF K=1 IS MADE.
910 CONTINUE
C
C...  TOP OF GRID.
IF(TOPBDY.NE.'WALL'.AND.TOPBDY.NE.'FREE')GO TO 920
C
NEUMANN CONDITION AT K=KMAX.
DO 912 L=1,IMAX
DO 912 J=1,JMAX
A(J,KMAX,L) = A(J,KMAX,L)
912 CONTINUE

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
JN(J,KMAX,L) = JN(J,KMAX,L)
OMEGA(J,KMAX,L) = OMEGA(J,KMAX,L)
912 CONTINUE
920 CONTINUE
C IF TOPBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY
C CONDITION IS ASSUMED AND NO MODIFICATION OF KMAXL IS MA
C
C... SIDE OF GRID
IF(SIDBDY.NE.'WALL'.AND.SIDBDY.NE.'FREE') GO TO 925
C NEUMANN CONDITIONS AT J=JMAXL.
DO 922 L=1,IMAX
DO 922 K=1,KMAXL
A(JMAXL,K,L) = A(JMAX,K,L)
JN(JMAXL,K,L) = JN(JMAX,K,L)
OMEGA(JMAXL,K,L) = OMEGA(JMAX,K,L)
922 CONTINUE
925 CONTINUE
C IF SIDBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
C IS ASSUMED AND NO MODIFICATION OF JMAXL IS MADE.
C... FINISHED BOUNDARY CONDITIONS ON RHO, A, JN, AND OMEGA.
RETURN
END
SUBROUTINE EOC(ISYM,BOTBDY,TOPBDY,SIDBDY)
FOR THE FULLY SECOND ORDER DIFFERENCE WITH EVERYTHING RHO, A, S,
CENTERED IN THE SAME PLACE AT THE CENTER OF A CELL.
C
PARAMETER (JMAX=64, JMAXL = JMAX+1, JMAX2 = JMAX+2,  
1 KMAX=16, KMAXL = KMAX+1, KMAX2 = KMAX+2,  
2 LMAX=64)
C
COMMON /BL0K6/ DTHETA, COSIGN(LMAX),SIGN(LMAX),PI,GRAV
COMMON /BL0K7/ RCPD,CONSP,DELT,BDTEM,RN,DEL,TIME,CORMAS
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAXL),  
1 R0F3N,Z0F3N,ALNEWR,ALNEWZ
REAL JN
C* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
C* EXCEPT RIGHT NOW JN IS USED IN DELTA TO CALCULATE THE TIMESTEP
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),  
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),  
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),  
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /STATES/ P(JMAX2,KMAX2,LMAX)
COMMON /NORMAL/ CIPR
COMMON /FREEZ/ DENEX
REAL JLOSTR,JLOSTZ,MLOSTR,MLOSTZ
COMMON /ANGMOM/ JLOSTR,JLOSTZ,MLOSTR,MLOSTZ
COMMON /GEX/ GEMS,GEMR,GEMZ(JMAX1),VOLMI(JMAX1),DT(JMAX1),  
1 RHFI(JMAX1),RHFI2(JMAX1),DR,DZ,DETHI,RHFI3(JMAX1)
COMMON /FLUX/ FR(JMAX1,KMAX1,LMAX),FZ(JMAX1,KMAX1,LMAX),  
1 FT(JMAX1,KMAX1,LMAX),UTIL(JMAX1,KMAX1,LMAX)
CHARACTER*6 BOTBDY,TOPBDY,SIDBDY
CONDITIONAL VECTOR MERGE FUNCTIONS DEFINED.

LOGICAL ILL, ILL2

ON THE FPS-264 AFIX IS WRITTEN AS:

AFIX(ILL) = FLOAT(AND(SHIFT(ILL, -31), 1))

AN ALTERNATE WAY TO WRITE AFIX ON THE FPS-264 IS:

AFIX(ILL) = FLOAT(SHIFT(ILL, -63))

THIS SAVES SOME TIME BUT ENGINEERING IS CHECKING OUT IF THIS WILL ALWAYS WORK. FOR LOGICAL .TRUE. THE FPS PUTS A 1 IN BIT 32

11. FOR .FALSE. THE FPS PUT A 0 IN BIT 32 AND 11. THE OTHER BI

C IN A LOGICAL ARE IGNORED.

ON A VAX USING VMS 4.5 AFIX CAN BE WRITTEN AS:

AFIX(ILL) = ABS(FLOAT(ILL))

THE VAX MAKES .TRUE. -1 AND .FALSE. 0.

ON AN IBM AFIX MUST BE MADE A FUNCTION BECAUSE OF THE SILLY WAY THE IBM COMPILER CHECKS VARIABLE TYPING AT COMPILE TIME.

FUNCTION AFIX(III)

AFIX = FLOAT(III)

RETURN

END

ON THE IBM OF COURSE THE FUNCTION AFIX(III) CAN NOT BE PLACED HE AT LEAST THAT IS THE WAY VSFORT WAS ON 3-9-87.

ON ANY CRAY MACHINE CVMGM, CVMGP, CVMGZ, CVMGN, AND CVMGT ARE AL DEFINED WITHIN THE FORTRAN LANGUAGE AS EXTENSIONS.

CVMGM(F, B, C) = AFIX(C .LT. 0.0) * F + AFIX(C .GE. 0.0) * B

CVMGP(F, B, C) = AFIX(C .GE. 0.0) * F + AFIX(C .LT. 0.0) * B

CVMGPl(F, B, C) = AFIX(C .GT. 0.0) * F + AFIX(C .LE. 0.0) * B

CVMGZ(F, B, C) = AFIX(C .EQ. 0.0) * F + AFIX(C .NE. 0.0) * B

CVMGN(F, B, C) = AFIX(C .NE. 0.0) * F + AFIX(C .EQ. 0.0) * B

CVMGT(F, ILL) = AFIX(ILL) * F + AFIX(.NOT. ILL) * B

CVMGT2(F, B, ILL2) = AFIX(ILL2) * F + AFIX(.NOT. ILL2) * B

CVMZPI(B) = CVMGT(0.0, 1.0, B .LE. 1.0E-100)/

1 CVMGT2(1.0, B, B .LE. 1.0E-100)

C------ VAN LEER SLOPES FOR A -------------------------------

SLOPEA(F, B, C) = CVMGT(0., 1., ABS(C-F) .LE. 1.E-100

1 .OR. (C-B)*(B-F) .LE. 0. )

2 *( 2.*(B-C)*(F-B) )/

3 CVMGZ(1., F-C,

4 ABS(C-F) .LE. 1.0E-100 .OR. (C-B)*(B-F) .LE. 0.0)

C------ FIRST ORDER SLOPES FOR A -----------------------------

SLOPEA(F, B, C) = 0.0

C------ VAN LEER SLOPES FOR RHO ------------------------------

SLOPER(F, B, C) = CVMGPl(1., 0., (B-C)*(F-B)) * 2.*(B-C)*(F-B)/

1 CVMGZ(1., F-C, F-C)

C------ FIRST ORDER SLOPES FOR RHO ----------------------------

SLOPER(F, B, C) = 0.0

C------------------------
C-----VAN LEER INTERPOLATION FOR RHO OR A OR EPS**{(1/GAMMA)}-------
C-IN THE J ,K OR L DIRECTION WHERE DX DOES NOT CHANGE WITH GRID CE
VLI(Q,Q1,DX,V,SLOPE,SLOPE1)=
  1 CVMGP(Q1+(DX-V*DELT)*SLOPE1/(2.*DX),
    Q -(DX+V*DELT)*SLOPE/(2.*DX),V)
C------------------------------------------------------------------
C------FIRST ORDER INTERPOLATION FOR RHO OR A OR EPS**{(1/GAMMA)}------
C-IN THE J ,K OR L DIRECTION WHERE DX DOES NOT CHANGE WITH GRID CE
VLI(Q,Q1,DX,V,SLOPE,SLOPE1)=CVMGP(Q1,Q,V)
C------------------------------------------------------------------

ISYMA=IABS(ISYM)
HALFDT = 0.5*DELT
C-----------VAN LEER SLOPES FOR A IN THE J DIRECTION ------------
DO 703 L=1,LMAX
  DO 702 K=2,KMAX
    DO 701 J=2,JMAX
      JP=J+1
      JM=J-1
      UTIL(J,K,L)=SLOPEA(A(JP,K,L),A(J,K,L),A(JM,K,L))
    701 CONTINUE
  702 CONTINUE
703 CONTINUE
C--------------FLUX'S FOR A IN THE J DIRECTION ---------------
DO 706 L=1,LMAX
  DO 705 K=2,KMAX
    FR(2,K,L)=0.0
    DO 704 J=3,JMAX
      JM=J-1
      TEMP=U(J,K,L)
      C-----------FLUX'S FOR A IN THE J DIRECTION USING R*(J) --------
      FR(J,K,L)=VLI(A(J,K,L),A(JM,K,L),DR,TEMP,
        UTIL(J,K,L),UTIL(JM,K,L))*TEMP*GEMR*(R(J)-TEMP*HALFDT)
      C----------FLUX'S FOR A IN THE J DIRECTION WITHOUT USING R*(J)------
      C FR(J,K,L)=VLI(A(J,K,L),A(JM,K,L),DR,TEMP,
        UTIL(J,K,L),UTIL(JM,K,L))*TEMP*GEMR*R(J)
    704 CONTINUE
  705 CONTINUE
706 CONTINUE
C----------------------------------------------------------------
C----------VAN LEER SLOPES FOR A IN THE K DIRECTION ---------
DO 709 L=1,LMAX
  DO 708 K=2,KMAX
    KP=K+1
    KM=K-1
    DO 707 J=2,JMAX
      UTIL(J,K,L)=SLOPEA(A(J,KP,L),A(J,K,L),A(J,KM,L))
    707 CONTINUE
  708 CONTINUE
709 CONTINUE
C--------------FLUX'S FOR A IN THE K DIRECTION ---------------
DO 712 L=1,LMAX
  Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 711 K=2,KMAX1
KM=K-1
DO 710 J=2,JMAX
TEMP=W(J,K,L)
FZ(J,K,L)=VLI(A(J,K,L),A(J,KM,L),DZ,TEMP,
1 UTIL(J,K,L),UTIL(J,KM,L))*TEMP*GEMZ(J)
710 CONTINUE
711 CONTINUE
712 CONTINUE

C-----------VAN LEER SLOPES FOR A IN THE L DIRECTION-----------
C******* FOR THE L=1 CASE SLOPES *******************************
L=1
LP=2
LM=LMAX
DO 714 K=2,KMAX
DO 713 J=2,JMAX
UTIL(J,K,L)=SLOPEA(A(J,K,LP),A(J,K,L),A(J,K,LM))
713 CONTINUE
714 CONTINUE

C******* FOR THE L=LMAX CASE SLOPES ******************************
L=LMAX
LP=1
LM=LMAX-1
DO 716 K=2,KMAX
DO 715 J=2,JMAX
UTIL(J,K,L)=SLOPEA(A(J,K,LP),A(J,K,L),A(J,K,LM))
715 CONTINUE
716 CONTINUE

C******* FOR THE L=2,LMAX-1 CASE SLOPES **************************
DO 719 L=2,LMAX-1
LP=L+1
LM=L-1
DO 718 K=2,KMAX
DO 717 J=2,JMAX
UTIL(J,K,L)=SLOPEA(A(J,K,LP),A(J,K,L),A(J,K,LM))
717 CONTINUE
718 CONTINUE
719 CONTINUE

C---------------------------------------------------------------
C--------------FLUX'S FOR A IN THE L DIRECTION------------------
C******* FOR THE L=1 CASE FLUX'S *******************************
L=1
LM=LMAX
DO 721 K=2,KMAX
DO 720 J=2,JMAX
TEMP=VT(J,K,L)
FT(J,K,L)=VLI(A(J,K,L),A(J,K,LM),DT(J),TEMP,
1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
720 CONTINUE
721 CONTINUE

C******* FOR THE L=2,LMAX CASE FLUX'S ***************************
DO 726 L=2,LMAX
LM=L-1
DO 725 K=2,KMAX

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DO 724 J=2,JMAX
  TEMP=VT(J,K,L)
  FT(J,K,L)=VLI(A(J,K,L),A(J,K,LM),DT(J),TEMP,
  1 UTIL(J,K,L),UTIL(J,K,LM))*TEMP*GEMS
724 CONTINUE
725 CONTINUE
726 CONTINUE
C------------------------------------------------------------------
C--------COMPUTE THE NEW A'S FROM THE FLUX'S----------------------
C******* FOR THE L=LMAX CASE THE NEW A'S************************
L=LMAX
  LP=1
  LM=LMAX-1
  DO 728 K=2,KMAX
    KP=K+1
    DO 727 J=2,JMAX
      JP=J+1
      1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
727 CONTINUE
728 CONTINUE
C******* FOR THE L=1 CASE THE NEW A'S************************
L=1
  LP=2
  LM=LMAX
  DO 730 K=2,KMAX
    KP=K+1
    DO 729 J=2,JMAX
      JP=J+1
      1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
729 CONTINUE
730 CONTINUE
C******* FOR THE L=2,LMAX-1 CASE THE NEW A'S*****************
DO 733 L=2,LMAX-1
  LP=L+1
  LM=L-1
  DO 732 K=2,KMAX
    KP=K+1
    DO 731 J=2,JMAX
      JP=J+1
      1 FT(J,K,LP)-FT(J,K,L))*VOLMI(J)*DELT
731 CONTINUE
732 CONTINUE
C... KEEP TRACK OF HOW MUCH ANG. MOM. ARE LOST THROUGH
C THE R AND Z GRID BOUNDARIES. JLOST IS NOT A SPECIFIC ANGULAR
C LOST.
DO 1245 L=1,LMAX
  DO 1244 K=2,KMAX
    JLOSTR = JLOSTR + DELT*FR(JMAX1,K,L)
1244 CONTINUE
1245 CONTINUE
DO 2245 L=1,LMAX
DO 2244 J=2,JMAX
JLOSTZ = JLOSTZ + DELT*FZ(J,KMAX1,L)
2244 CONTINUE
2245 CONTINUE

C********NOW DO THE CONTINUITY EQUATION**************************
C
-------------------VAN LEER SLOPES FOR RHO IN THE J DIRECTION-------------------
DO 803 L=1,LMAX
DO 802 K=2,KMAX
DO 801 J=2,JMAX1
JP=J+1
JM=J-1
UTIL(J,K,L)=SLOPER(RHO(JP,K,L),RHO(J,K,L),RHO(JM,K,L))
801 CONTINUE
802 CONTINUE
803 CONTINUE
C------------------------------------------------------------------
C FLUX'S FOR RHO IN THE J DIRECTION ------------------
DO 806 L=1,LMAX
DO 805 K=2,KMAX
C FR(2,K,L)=0.0 IS ALREADY SET WHEN FLUX'S FOR A IN THE J DIRECT
C FR(2,K,L)=0.0
DO 804 J=3,JMAX1
JM=J-1
TEMP=U(J,K,L)
C-------FLUX'S FOR RHO IN THE J DIRECTION USING R*(J) -------------
FR(J,K,L)=VLI(RHO(J,K,L),RHO(JM,K,L),DR,TEMP,
1 UTIL(J,K,L),UTIL(JM,K,L))*TEMP*GEMR*(R(J)-TEMP*HALFDT)
C-------FLUX'S FOR RHO IN THE J DIRECTION WITHOUT USING R*(J) ----
C FR(J,K,L)=VLI(RHO(J,K,L),RHO(JM,K,L),DR,TEMP,
C 1 UTIL(J,K,L),UTIL(JM,K,L))*TEMP*GEMR*R(J)
C------------------------------------------------------------------
804 CONTINUE
805 CONTINUE
806 CONTINUE
C------------------------------------------------------------------
C-----------VAN LEER SLOPES FOR RHO IN THE K DIRECTION -----------
DO 809 L=1,LMAX
DO 808 K=2,KMAX1
KP=K+1
KM=K-1
DO 807 J=2,JMAX
UTIL(J,K,L)=SLOPER(RHO(J,KP,L),RHO(J,K,L),RHO(J,KM,L))
807 CONTINUE
808 CONTINUE
809 CONTINUE
C------------------------------------------------------------------
C-------------------VAN LEER SLOPES FOR RHO IN THE K DIRECTION-------------------
DO 812 L=1,LMAX
DO 811 K=2,KMAX1
KM=K-1
DO 810 J=2,JMAX
TEMP=W(J,K,L)
\[ FZ(J,K,L) = VLI(RHO(J,K,L), RHO(J,KM,L), DZ, TEMP, \\
1 \text{UTIL}(J,K,L), \text{UTIL}(J,KM,L)) \times TEMP \times GEMZ(J) \]

810 CONTINUE
811 CONTINUE
812 CONTINUE

C-----------VAN LEER SLOPES FOR RHO IN THE L DIRECTION-----------
C******** FOR THE L=1 CASE FLUX'S ****************************
L=1
LP=2
LM=LMAX
DO 814 K=2,KMAX
DO 813 J=2,JMAX
\text{UTIL}(J,K,L) = SLOPER(RHO(J,K,LP), RHO(J,K,L), RHO(J,K,LM))
813 CONTINUE
814 CONTINUE

C******** FOR THE L=LMAX CASE FLUX'SX ****************************
L=LMAX
LP=1
LM=LMAX-1
DO 816 K=2,KMAX
DO 815 J=2,JMAX
\text{UTIL}(J,K,L) = SLOPER(RHO(J,K,LP), RHO(J,K,L), RHO(J,K,LM))
815 CONTINUE
816 CONTINUE

C******** FOR THE L=2,LMAX-1 CASE FLUX'S ****************************
DO 819 L=2,LMAX-1
LP=L+1
LM=L-1
DO 818 K=2,KMAX
DO 817 J=2,JMAX
\text{UTIL}(J,K,L) = SLOPER(RHO(J,K,LP), RHO(J,K,L), RHO(J,K,LM))
817 CONTINUE
818 CONTINUE
819 CONTINUE

C-----------------FLUX'S FOR RHO IN THE L DIRECTION-----------------
C******** FOR THE L=1 CASE FLUX'S ****************************
L=1
LM=LMAX
DO 821 K=2,KMAX
DO 820 J=2,JMAX
\text{TEMP} = \text{VT}(J,K,L)
\text{FT}(J,K,L) = VLI(RHO(J,K,L), RHO(J,K,LM), DT(J), TEMP, \\
1 \text{UTIL}(J,K,L), \text{UTIL}(J,K,LM)) \times TEMP \times GEMS
820 CONTINUE
821 CONTINUE

C******** FOR THE L=2,LMAX CASE FLUX'S ****************************
DO 826 L=2,LMAX
LM=L-1
DO 825 K=2,KMAX
DO 824 J=2,JMAX
\text{TEMP} = \text{VT}(J,K,L)
\text{FT}(J,K,L) = VLI(RHO(J,K,L), RHO(J,K,LM), DT(J), TEMP, \\
1 \text{UTIL}(J,K,L), \text{UTIL}(J,K,LM)) \times TEMP \times GEMS

C------------COMPUTE THE NEW RHO'S FROM THE FLUX'S----------------
C******* FOR THE L=LMAX CASE THE NEW RHO'S************************
L=LMAX
LP=1
DO 828 K=2,KMAX
KP=K+1
DO 827 J=2,JMAX
JP=J+1
RHO(J,K,L)=RHO(J,K,L)-(FR(JP,K,L)-FR(J,K,L)+FZ(J,KP,L)-FZ(J,
1 FT(J,K,LP)-FT(J,K,L))*DELT*VOLMI(J)
827 CONTINUE
828 CONTINUE
C******* FOR THE L=1,LMAX-1 CASE THE NEW RHO'S********************
DO 831 L=1,LMAX-1
LP=L+1
DO 830 K=2,KMAX
KP=K+1
DO 829 J=2,JMAX
JP=J+1
RHO(J,K,L)=RHO(J,K,L)-(FR(JP,K,L)-FR(J,K,L)+FZ(J,KP,L)-FZ(J,
1 FT(J,K,LP)-FT(J,K,L))*DELT*VOLMI(J)
829 CONTINUE
830 CONTINUE
831 CONTINUE
C... KEEP TRACK OF HOW MUCH MASS IS LOST THROUGH
C THE R AND Z GRID BOUNDARIES.
DO 1247 L=1,LMAX
DO 1246 K=2,KMAX
MLOSTR = MLOSTR + DELT*FR(JMAX1,K,L)
1246 CONTINUE
1247 CONTINUE
DO 2247 L=1,LMAX
DO 2246 J=2,JMAX
MLOSTZ = MLOSTZ + DELT*FZ(J,KMAX1,L)
2246 CONTINUE
2247 CONTINUE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CC CC
CC FINISHED MAJOR WORK ... NOW CLEAN UP CC
CC CC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C... SMOOTH A(J,K,L) OVER J=2 AND 3 TO PRODUCE UNIFORM ROTATION
C OVER THESE ZONES.
RR2=RHF(2)**2
RR3=RHF(3)**2
ARSMO2=R(3)*R(3)
ARSMO3=R(4)*R(4) - ARSMO2
ARRAT=ARSMO3/ARSMO2
DO 300 K=2,KMAX1
300 CONTINUE
R2=0.0
XR3=0.0
A2=0.0
A3=0.0
DO 301 L=1,LMAX
R2=R2 + RH0(2,K,L)
XR3=XR3 + RH0(3,K,L)
A2=A2 + A(2,K,L)
A3=A3 + A(3,K,L)
301 CONTINUE
RSQM=R2*RR2 + ARRAT*XR3*RR3
ATOT=A2 + ARRAT*A3
OMUR=ATOT/RSQM
DO 302 L=1,LMAX
A(2,K,L)=RR2*OMUR*RHO(2,K,L)
A(3,K,L)=RR3*OMUR*RHO(3,K,L)
302 CONTINUE
300 CONTINUE
C... REDEFINE OMEGA AND JN.
DO 500 L = 1,LMAX
DO 500 K=2,KMAX
DO 500 J=2,JMAX1
JN(J,K,L) = A(J,K,L)/RHO(J,K,L)
C* OMEGA(J,K,L) = JN(J,K,L) * RHFINV(J)
OMEGA(J,K,L) = JN(J,K,L) * RHFI2(J)
500 CONTINUE
C... TO PREVENT TIME STEP FROM BEING CONTROLLED BY HIGH ANGULAR VELOCITIES IN LOW DENSITY REGIONS, SET JN AND OMEGA TO ZERO
C IF RHO.LE.D AND IF OMEGA.GT.FUDGE*(CENTRAL OMEGA).
D=1.0E3*DENEX
FUDGE = 0.25
ROTMAX = 2.0*PI/CIRP*FUDGE
DO 600 L=1,LMAX
DO 600 K=2,KMAX1
DO 600 J=2,JMAX1
IF(RHO(J,K,L).GT.D)GO TO 600
IF(ABS(OMEGA(J,K,L)).LE.ROTMAX)GO TO 600
JN(J,K,L)=0.
OMEGA(J,K,L)=0.0
A(J,K,L)=0.0
VT(J,K,L)=0.0
600 CONTINUE
C C C
C C C
C... SET BOUNDARY CONDITIONS ON RHO, A, JN, AND OMEGA.
C C
C... Z-AXIS.
C NEUMANN CONDITION HOLDS UNLESS ISYMA = 1 OR 2.
LHAF=LMAX/2
LSTOP=LHAF
IF(ISYMA.NE.1.AND.ISYMA.NE.2)LSTOP=LMAX
DO 292 L=1,LSTOP
LP=L+LHAF
IF( LSTOP.EQ.1MAX) LP=L
DO 292 K=2,KMAX
RH0(1,K,L) = RH0(2,K,LP)
RH0(1,K,LP) = RH0(2,K,L)
A(1,K,L) = A(2,K,LP)
A(1,K,LP) = A(2,K,L)
JN(1,K,L) = JN(2,K,LP)
JN(1,K,LP) = JN(2,K,L)
OMEGA(1,K,L) = OMEGA(2,K,LP)
OMEGA(1,K,LP) = OMEGA(2,K,L)
292 CONTINUE
C... BOTTOM OF GRID.
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8) GO TO 296
C NEUMANN CONDITION IF SYMMETRY THRU EQUATORIAL PLANE ASS
294 DO 295 L=1,LMAX
DO 295 J=1,JMAX1
RH0(J,1,L) = RH0(J,2,L)
A(J,1,L) = A(J,2,L)
JN(J,1,L) = JN(J,2,L)
OMEGA(J,1,L) = OMEGA(J,2,L)
295 CONTINUE
GO TO 910
296 CONTINUE
C IF BOTBDY = 'WALL' OR 'FREE', THEN REFLECTION SYMMETRY
C AMOUNTS TO THE SAME THING AS NEUMANN CONDITION.
IF(BOTBDY.EQ.'WALL'.OR.BOTBDY.EQ.'FREE') GO TO 294
C IF BOTBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
C IS ASSUMED AND NO MODIFICATION OF i:=1 IS MADE.
910 CONTINUE
C
C... TOP OF GRID.
IF(TOPBDY.NE.'WALL'.AND.TOPBDY.NE.'FREE')GO TO 920
C NEUMANN CONDITION AT K=KMAX1.
DO 912 L=1,LMAX
DO 912 J=1,JMAX1
RH0(J,KMAX1,L) = RHO(J,KMAX,L)
A(J,KMAX1,L) = A(J,KMAX,L)
JN(J,KMAX1,L) = JN(J,KMAX,L)
OMEGA(J,KMAX1,L) = OMEGA(J,KMAX,L)
912 CONTINUE
920 CONTINUE
C IF TOPBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
C CONDITION IS ASSUMED AND NO MODIFICATION OF KMAX1 IS MA
C
C... SIDE OF GRID
IF(SIDBDY.NE.'WALL'.AND.SIDBDY.NE.'FREE')GO TO 925
C NEUMANN CONDITIONS AT J=JMAX1.
DO 922 L=1,LMAX
DO 922 K=1,KMAX1
RH0(JMAX1,K,L) = RHO(JMAX,K,L)
A(JMAX1,K,L) = A(JMAX,K,L)
JN(JMAX1,K,L) = JN(JMAX,K,L)
OMEGA(JMAX1,K,L) = OMEGA(JMAX,K,L)
922 CONTINUE

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
CONTINUE
IF SIDBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY IS ASSUMED AND NO MODIFICATION OF JMAX1 IS MADE.

FINISHED BOUNDARY CONDITIONS ON RHO, EPS, A, JN, AND OMEGA.

RETURN
END

SUBROUTINE DELTA

EVENTUALLY JN AND OMEGA SHOULD BE REMOVED SINCE THEY ARE UNNECESSARY AND USE LOTS OF STORAGE BEING 3-D ARRAYS.

PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)

COMMON /BL0K6/ DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAY
COMMON /BL0K7/ RCLOUD,CONSTP,DELT,BDYTEM,DEN,TIME,CORMAS
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
1 R0F3N, Z0F3N, AlNEWR, AINEWZ
REAL JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /STATES/ P(JMAX2,KMAX2,IMAX)
COMMON /TIMST/ INDX,ISOADI,ALLOW,ITSTEP
REAL KONST
COMMON /PTROPE/ XN,XN1,KONST
COMMON /QRR/ QRR(JMAX,KMAX,LMAX),QZZ(JMAX,KMAX,LMAX),
1 QT(T(JMAX,KMAX,LMAX))
COMMON /GEX/ GEMS,GEMR,GEMZ(JMAX1),VOLMI(JMAX1),DT(JMAX1),
1 RHFI(JMAX1),RHFI2(JMAX1),DR,DZ,DELT,HIF,HIFI3(JMAX1)
DIMENSION SP(4)

FACTOR IS FRACTION OF COURANT TIME BEING USED.
ALLOW IS MAXIMUM ALLOWABLE FRACTIONAL CHANGE IN MAXIMUM DENSITY DURING ONE TIME STEP. DMAXO WILL BE STORED IN RHO(JMAX2, KMAX2, LMAX)
FACTOR=0.7
ALLOW = 0.15
GAMMA = XN1
ADISO=1.0
IF(ISOADI.EQ.2.OR. ISOADI.EQ.3) ADISO=GAMMA
DMAXO=RHO(JMAX2, KMAX2, LMAX)
AMIN=.1E-15
DO 90 L=1,LMAX
DO 90 K=2,KMAX
DO 90 J=3,JMAX
SPEED1=SQRT(ADISO*P(J,K,L)/RHO(J,K,L))
SPEED2=ABS(U(J,K,L)) + SQRT(QRR(J,K,L)/RHO(J,K,L))*4.0
SPEED3 = ABS(W(J,K,L)) + SQRT(QZZ(J,K,L)/RHO(J,K,L)) * 4.0
SPEED4 = ABS(VT(J,K,L)) + SQRT(QTT(J,K,L)/RHO(J,K,L)) * 4.0
DELT1 = DR/(SPEED1 + SPEED2)
DELT2 = DZ/(SPEED1 + SPEED3)
DELT3 = DT(J)/(SPEED1 + SPEED4)
DELT1 = AMIN1(DELT1, DELT2)
DELT1 = AMIN1(DELT1, DELT3)
IF(DELT1 < AMIN) GO TO 53
AMIN = DELT1
J1 = J
K1 = K
L1 = L
SP(1) = SPEED1
SP(2) = SPEED2
SP(3) = SPEED3
SP(4) = SPEED4
90 CONTINUE
DELT = FACTOR*AMIN
C... NOW CHOP THIS COURANT-DETERMINED TIME BY DENSITY CHANGE CRIT
DMAX = RHO(2,2,1)
DO 95 L=1,LMAX
   DO 95 K=2,KMAX
      DO 95 J=2,JMAX
         IF(RHO(J,K,L).LE.DMAX) GO TO 95
      DMAX = RHO(J,K,L)
95 CONTINUE
CHGMAX = ABS(1.0-DMAXO/DMAX)
IF(INDX.LE.1.AND.DMAXO.LT.DEN) GO TO 98
F1 = ALLOW/CHGMAX
F2 = ALLOW/RHO(JMAX2,KMAX1,L)
CHGMAX = CHGMAX
DELT = DELT*AMIN1(1.0,F1)*AMIN1(1.0,F2)
98 RHO(JMAX2,KMAX2,L) = DMAX
100 WRITE(7,100) DELT, J1, K1, L1
   FORMAT(' DELT=',1PE13.5,' J1=',I2, ' K1=',I2, ' L1=',I2)
C WRITE(7,101) SP
   FORMAT(' SP=',1P4E13.5)
   WRITE(7,102) ALLOW, CHGMAX
C WRITE(7,103) TIME = TIME+DELT
RETURN
END
SUBROUTINE VLIMIT
C
PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
           KM=16, KM1 = KM+1, KM2 = KM+2,
           LMAX=64)
C
COMMON /BLOK7/ RCLCLOUD, CONSTP, DELT, BDYTEM, DEN, TIME, CORMAS
   REAL JN
COMMON /EOM/ S(JMAX2,KMAX2,LMAX), T(JMAX2,KMAX2,LMAX),
               A(JMAX2,KMAX2,LMAX), U(JMAX2,KMAX2,LMAX),
               W(JMAX2,KMAX2,LMAX), JN(JMAX2,KMAX2,LMAX),
               OMEGA(JMAX2,KMAX2,LMAX), VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX), RHO(JMAX2,KMAX2,LMAX)
COMMON /TIMST/ INDX, ISOADI, ALLOW, ITSTEP
COMMON /VEIMAX/ VMXIN, VMXOUT
REAL KONST
COMMON /PTROPE/ XN, GAMMA, KONST
COMMON /FREEZ/ DENEX
DIMENSION USIGN(JMAX2)

DIMENSION PHI( JMAX2, KMAX2, LMAX), RHO( JMAX2, KMAX2, LMAX)
INDX, ISOADI, ALLOW, ITSTEP
VMXIN, VMXOUT

KONST XN, GAMMA, KONST
DENEX
USIGN(JMAX2)

SET D = FACTOR*DENEX, WHERE DENEX (= 1.E-7*DEN, TYPICALLY)
IS THE INITIAL VALUE OF BACKGROUND DENSITY.
THEN LIMIT U, W, S, AND T DIFFERENTLY INSIDE AND OUTSIDE
OF THE GAS CLOUD, DEPENDING ON IF RHO-AVERAGE IS GREATER
THAN OR LESS THAN, RESPECTIVELY, THE VALUE OF D.

VMXIN AND VMXOUT ARE READ IN FROM UNIT 5 INSIDE SETUP.
BUT, IF ISOADI=3, VMXOUT IS OVERRULED BY THE VALUE OF
VLIM = 0.5*(SOUND SPEED AT MAXIMUM DENSITY).

CHECK THE FOLLOWING...
1. ARRAY 'USIGN' DETERMINES THE SIGN GIVEN TO VLIM;
   IT IS +1 IF J.LE.JDECID,
   IT IS -1 IF J.GT.JDECID.
2. JDECID = 41; THIS VALUE MAY NEED TO BE CHANGED.
3. FACTOR = 500.0

DATA JDECID/41/
DATA FACTOR/500.0/

IF(JDECID.LT.1)JDECID=1
IF(JDECID.GT. JMAX) JDECID=JMAX
D = FACTOR*DENEX
DMAX=AMAX1(DEN, RHO(JMAX2,KMAX2,1))
VLIM=0.5*SQRT(GAMMA*(DMAX**(GAMMA-1.)))
IF(ISOADI.NE.3)VLIM=VMXOUT

INITIALIZE ARRAY 'USIGN'
DO 703 J=1, JDECID
703 USIGN(J) = 1.0
JNEXT = JDECID + 1
DO 704 J = JNEXT, JMAX2
704 USIGN(J) = -1.0

LIMIT THE ALLOWED MAXIMUM VELOCITIES
DO 701 L=1, LMAX
   IM=L-1
   IF(LMAX.EQ.1)IM=L
   IF(IM.EQ.0)IM=LMAX
   DO 701 K=2, KMAX1
      KM=K-1
      DO 701 J=2, JMAX1
         JM=J-1
         RHJKL=RHO(J,K,L)
IF(RHJKL.LT.D)GO TO 755

C.....INSIDE CLOUD............
IF(ABS(U(J,K,L)).LE.VMXIN)GO TO 711

C LIMIT U AND S................
U(J,K,L)=VMXIN*U(J,K,L)/ABS(U(J,K,L))
S(J,K,L) = U(J,K,L)*RHJKL*2.0-S(JM,K,L)*RHJKL/RHO(JM,K,L)
711 IF(ABS(W(J,K,L)).LE.VMXIN)GO TO 785

C LIMIT W AND T................
W(J,K,L)=VMXIN*W(J,K,L)/ABS(W(J,K,L))
GO TO 785

755 CONTINUE

C.....OUTSIDE CLOUD............
IF(ABS(U(J,K,L)).LE.VLIM)GO TO 761

C LIMIT U AND S................
U(J,K,L) = VLIM*USIGN(J)
S(J,K,L) = U(J,K,L)*RHJKL*2.0-S(JM,K,L)*RHJKL/RH0(JM,K,L)
761 IF(ABS(W(J,K,L)).LE.VLIM)GO TO 785

C LIMIT W AND T................
W(J,K,L) = - VLIM
785 CONTINUE

701 CONTINUE
RETURN
END

SUBROUTINE STATE(I,J,K,L)

C
PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64, MUCH=JMAX2*KMAX2*LMAX)
C
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /STATES/ P(JMAX2,KMAX2,LMAX)
C* 1 EPS(JMAX2,KMAX2,LMAX)
REAL KONST
COMMON /PTROPE/ XN,XN1,KONST
COMMON /BL0K7/ RCLOUD,CONSTP,DELT,BDYTEM,DEN,TIME,CORMAS
COMMON /STEP/ ICNT
C
C... POLYTROPIC RELATION.
C DO 60 LL=1,LMAX
C DO 60 KK=1,KMAX1
C DO 60 JJ=1,JMAX1
C IF (RHO(JJ,KK,LL) .LT. 0.0) THEN
C WRITE(7,100)JJ,KK,LL,RHO(JJ,KK,LL),ICNT,TIME,DELT
100 FORMAT( '  NEGATIVE RHO(',12,',' ,12,,12,')  = ',1PE12.4,5X,
C 1 ' TIMESTEP#=',15,5X,' TIME=',1PE12.4,5X,' DELT=',1PE12.4)
C RHO(JJ,KK,LL)=ABS(RHO(JJ,KK,LL))
C ELSE
C ENDIF
C P(JJ,KK,LL) = KONST*RHO(JJ,KK,LL)**XN1
C 60 CONTINUE
C FPS IMPLEMENTATION OF P=RHO**GAMMA EXCEPT 0NOT=0**GAMMA

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
C  AND  NEGATIVE  RHO  NOT  TRAPPED.
C  CALL  VALG(RHO,1,P,1,MUCH)
C  CALL  VSMUL(P,1,XN1,P,1,MUCH)
C  CALL  VEXP(P,1,P,1,MUCH)
C  IF  POLYTROPIC  UNIT  ARE  BEING  USED  M=G=K=1,  THEN  SINCE  KONST=1
C  THE  FOLLOWING  CALL  NEED  NOT  BE  MADE.
C  CALL  VSMUL(P,1,KONST,P,1,MUCH)
C  CALL  VRPOW(RHO,  1, XN1, KONST, P, 1, MUCH, MVAL, MIDX, IERR)
IF(IERR .EQ. 1) THEN
   M1=0
   DO  12 L=1,LMAX
   DO  13 K=1,KMAX2
   DO  14 J=1,JMAX2
   MI=MI+1
   IF (MI .EQ. MIDX) THEN
      JJ=J
      KK=K
      LL=L
      ELSE
      ENDIF
   14 CONTINUE
   13 CONTINUE
   12 CONTINUE
WRITE(6,200)VAL,JJ,KK,LL
200 FORMAT( ' MINIMUM VALUE OF RHO IS ',1PE12.5,' AT J=',I3,2X, ' AT K=',I2,2X, ' AT L=',I3)
WRITE(6,*)'MIDX=',MIDX
ELSE
ENDIF
RETURN
END

SUBROUTINE VIS
C
C  ARTIFICIAL  QUADRATIC  VISCOSITY.  WITH  QRR,  QZZ,  AND  QTT  SAVED  IN
C  A  COMMON  BLOCK  CALLED  QS.
C
PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
   1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
   2 LMAX=64)
C
COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAV
COMMON /NORMAL/ CIRP
COMMON /FREEZ/ DENEX
COMMON /BLOK7/ RCL OUD, CONSTP, DELT, BDXTEM, DEN, TIME, CORMAS
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX1),
   1 ROF3N,ZOF3N,A1NEWR,A1NEWZ
REAL  JN
C*  JN  AND  OMEGA  ARE  EXTRA  USELESS  ARRAYS  IN  THIS  SUBROUTINE.
C*  EXCEPT  RIGHT  NOW  JN  IS  USED  IN  DELTA  TO  CALCULATE  THE  TIMESTEP
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
   1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
   2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
   3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /GEX/ GEMS, GEMR, GEMZ(JMAX1), VOLMI(JMAX1), DT(JMAX1),
1 RHFI(JMAX1), RHFI2(JMAX1), DR, DZ, DELTHI, RHFI3(JMAX1)
CHARACTER*6 BOTBDY, TOPBDY, SIDBDY
COMMON /QS/ QRR(JMAX, KMAX, LMAX), QZZ(JMAX, KMAX, LMAX),
1 QTT(JMAX, KMAX, LMAX)

C------------------CONDITIONAL VECTOR MERGE FUNCTIONS DEFINED.------------------
LOGICAL ILL, ILL2
C ON THE FPS-264 AFIX IS WRITTEN AS:
AFIX(ILL)=FLOAT(AND(SHIFT(ILL,-31),1))
C AN ALTERNATE WAY TO WRITE AFIX ON THE FPS-264 IS:
AFIX(ILL)=FLOAT(SHIFT(ILL,-63))
C THIS SAVES SOME TIME BUT ENGINEERING IS CHECKING OUT IF THIS
C WILL ALWAYS WORK. FOR LOGICAL .TRUE. THE FPS PUTS A 1 IN BIT 32
C 11. FOR .FALSE. THE FPS PUT A 0 IN BIT 32 AND 11. THE OTHER BI
C IN A LOGICAL ARE IGNORED.
C
C ON A VAX USING VMS 4.5 AFIX CAN BE WRITTEN AS:
AFIX(ILL)=ABS(FLOAT(ILL))
C THE VAX MAKES .TRUE. -1 AND .FALSE. 0.
C
C ON AN IBM AFIX MUST BE MADE A FUNCTION BECAUSE OF THE SILLY WAY
C THE IBM COMPILER CHECKS VARIABLE TYPING AT COMPILE TIME.
C
FUNCTION AFIX(I)
AFIX=FLOAT(I)
RETURN
END
C ON THE IBM OF COURSE THE FUNCTION AFIX(I) CAN NOT BE PLACED HE
C AT LEAST THAT IS THE WAY VSFORT WAS ON 3-9-87.
C
C ON ANY CRAY MACHINE CVMGM, CVMGP, CVMGZ, CVMGN, AND CVMGT ARE AL
C DEFINED WITHIN THE FORTRAN LANGUAGE AS EXTENSIONS.
CVMGM(F,B,C)=AFIX(C .LT. 0.0)*F+AFIX( C .GE. 0.0)*B

C------------------INITIALIZE A ZERO QZZ------------------
DO 801 L=1,LMAX
  DO 800 J=2,JMAX
    QZZ(J,1,L)=0.0
  800 CONTINUE
  801 CONTINUE
C
C FOR THE L=1 TO LMAX-1 CASE----------------------------------
DO 706 L=1,LMAX-1
  LP=L+1
  DO 705 K=2,KMAX
    KP=K+1
    QRR(1,K,L)=0.0
  705 CONTINUE
  DO 704 J=4,JMAX
    JP=J+1
    TEMP1=U(JP,K,L)-U(J,K,L)
    QRR(J,K,L)=RHO(J,K,L)*CVMGM(TMP1**2,0,TEMPI)
    TEMP2=W(J,KP,L)-W(J,K,L)
    QZZ(J,K,L)=RHO(J,K,L)*CVMGM(TEMP2**2,0,TEMP2)
    TEMP3=VT(J,K,LP)-VT(J,K,L)
  704 CONTINUE

QTT(J,K,L)=RHO(J,K,L)*CVMGM(TEMP3**2,0,TEMP3)

C FOR THE L=MAX -------------------------------------------------
L=LMAX
LP=1
DO 711 K=2,KMAX
KP=K+1
QRR(1,K,L)=0.0
DO 710 J=4,JMAX
JP=J+1
TEMP1=U(JP,K,L)-U(J,K,L)
QRR(J,K,L)=RHO(J,K,L)*CVMGM(TEMP1**2,0,TEMPS)
TEMP2=W(J,KP,L)-W(J,K,L)
QZZ(J,K,L)=RHO(J,K,L)*CVMGM(TEMP2**2,0,TEMPS)
TEMP3=VT(J,K,LP)-VT(J,K,L)
QTT(J,K,L)=RHO(J,K,L)*CVMGM(TEMP3**2,0,TEMPS)
710 CONTINUE
711 CONTINUE
C------------------------------------------------------------------
C COMPUTE THE NEW S, T, A FROM THE GRADIENT IN Q'S ----------------
C******* FOR THE L=1 CASE THE NEW S, T, AND A'S******************
L=1
LM=LMAX
DO 730 K=2,KMAX
KM=K-1
DO 729 J=5,JMAX
JM=J-1
S(J,K,L)=S(J,K,L)-(QRR(J,K,L)-QRR(JM,K,L))*DELT/DR
T(J,K,L)=T(J,K,L)-(QZZ(J,K,L)-QZZ(J,KM,L))*DELT/DZ
729 CONTINUE
730 CONTINUE
C******* FOR THE L=2,LMAX CASE THE NEW S, T, AND A'S**************
DO 733 L=2,LMAX
LM=L-1
DO 732 K=2,KMAX
KM=K-1
DO 731 J=5,JMAX
JM=J-1
S(J,K,L)=S(J,K,L)-(QRR(J,K,L)-QRR(JM,K,L))*DELT/DR
T(J,K,L)=T(J,K,L)-(QZZ(J,K,L)-QZZ(J,KM,L))*DELT/DZ
731 CONTINUE
732 CONTINUE
733 CONTINUE
CC FINISHED MAJOR WORK ... NOW CLEAN UP CC
CC CC CC
C... SMOOTH A(J,K,L) OVER J=2 AND 3 TO PRODUCE UNIFORM ROTATION
OVER THESE ZONES.
RR2=RHF(2)**2
RR3=RHF(3)**2
ARSMO2=R(3)*R(3)
ARSMO3=R(4)*R(4) - ARSMO2
ARRAT=ARSMO3/ARSMO2
DO 300 K=2,KMAX1
R2=0.0
XR3=0.0
A2=0.0
A3=0.0
DO 301 L=1,LMAX
R2=R2 + RHO(2,K,L)
XR3=XR3 + RHO(3,K,L)
A2=A2 + A(2,K,L)
A3=A3 + A(3,K,L)
301 CONTINUE
RSQM=R2*RR2 + ARRAT*XR3*RR3
ATOT=A2 + ARRAT*A3
OMUR=ATOT/RSQM
DO 302 L=1,LMAX
A(2,K,L)=RR2*OMUR*RHO(2,K,L)
A(3,K,L)=RR3*OMUR*RHO(3,K,L)
302 CONTINUE
300 CONTINUE
C... REDEFINE OMEGA AND JN.
DO 500 L = 1,LMAX
DO 500 K=2,KMAX
DO 500 J=2,JMAX1
JN(J,K,L) = A(J,K,L)/RHO(J,K,L)
C* OMEGA(J,K,L) = JN(J,K,L) * RHFINV(J)
OMEGA(J,K,L) = JN(J,K,L) * RHFI2(J)
500 CONTINUE
C... TO PREVENT TIME STEP FROM BEING CONTROLLED BY HIGH ANGULAR VELOCITIES IN LOW DENSITY REGIONS, SET JN AND OMEGA TO ZERO
C IF RHO.LE.D AND IF OMEGA.GT.FUDGE*(CENTERAL OMEGA).
D=1.0E3*DENEX
FUDGE = 0.25
ROTMAX = 2.0*PI/CIRP*FUDGE
DO 600 L=1,LMAX
DO 600 K=2,KMAX1
DO 600 J=2,JMAX1
IF(RHO(J,K,L).GT.D)GO TO 600
IF(ABS(OMEGA(J,K,L)).GT.ROTMAX)GO TO 600
JN(J,K,L)=0.0
OMEGA(J,K,L)=0.0
A(J,K,L)=0.0
VT(J,K,L)=0.0
600 CONTINUE

C... SET BOUNDARY CONDITIONS ON RHO, EPS, A, JN, AND OMEGA.
C. ... Z-AXIS.
C NEUMANN CONDITION HOLDS UNLESS ISYMA = 1 OR 2.
LHAF=LMAX/2
LSTOP=LHAF
IF(ISYMA.LE.1.AND.ISYMA.LE.2) LSTOP=LMAX
DO 292 L=1,LSTOP
LP=L+LHAF
IF(LSTOP.EQ.LMAX) LP=L
DO 292 K=2,KMAX
RH0(1,K,L) = RH0(2,K,LP)
RH0(1,K,LP) = RH0(2,K,L)
A(1,K,L) = A(2,K,LP)
A(1,K,LP) = A(2,K,L)
JN(1,K,L) = JN(2,K,LP)
JN(1,K,LP) = JN(2,K,L)
OMEGA(1,K,L) = OMEGA(2,K,LP)
OMEGA(1,K,LP) = OMEGA(2,K,L)
S(1,K,L) = S(2,K,LP)
S(1,K,LP) = S(2,K,L)
T(1,K,L) = T(2,K,LP)
T(1,K,LP) = T(2,K,L)
292 CONTINUE
C. ... BOTTOM OF GRID.
IF(ISYMA.EQ.1.OR.ISYMA.EQ.2) GO TO 296
C NEUMANN CONDITION IF SYMMETRY THRU EQUATORIAL PLANE ASS
294 DO 295 L=1,LMAX
DO 295 J=1,JMAX1
RH0(J,1,L) = RH0(J,2,L)
A(J,1,L) = A(J,2,L)
JN(J,1,L) = JN(J,2,L)
OMEGA(J,1,L) = OMEGA(J,2,L)
S(J,1,L) = S(J,2,L)
T(J,1,L) =-T(J,2,L)
295 CONTINUE
GO TO 910
296 CONTINUE
C IF BOTBDY = 'WALL' OR 'FREE', THEN REFLECTION SYMMETRY
C AMOUNTS TO THE SAME THING AS NEUMANN CONDITION.
IF(BOTBDY.EQ.'WALL'.OR.BOTBDY.EQ.'FREE') GO TO 294
C IF BOTBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
C IS ASSUMED AND NO MODIFICATION OF K=1 IS MADE.
910 CONTINUE
C. ... TOP OF GRID.
C IF(TOPBDY.NE.'WALL'.AND.TOPBDY.NE.'FREE') GO TO 920
C NEUMANN CONDITION AT K=KMAX1.
C DO 912 L=1,LMAX
C DO 912 J=1,JMAX1
C RH0(J,KMAX1,L) = RH0(J,KMAX,L)
C A(J,KMAX1,L) = A(J,KMAX,L)
C JN(J,KMAX1,L) = JN(J,KMAX,L)
C OMEGA(J,KMAX1,L) = OMEGA(J,KMAX,L)
912 CONTINUE
C 920 CONTINUE
IF TOPBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY
CONDITION IS ASSUMED AND NO MODIFICATION OF KMAX1 IS MA
C... SIDE OF GRID
IF(SIDBDY.NE.'WALL'.AND.SIDBDY.NE.'FREE')GO TO 925
NEUMANN CONDITIONS AT J=JMAX1.
DO 922 L=1,LMAX
DO 922 K=1,KMAX1
RHO(JMAX1,K,L) = RHO(JMAX,K,L)
A(JMAX1,K,L) = A(JMAX,K,L)
JN(JMAX1,K,L) = JN(JMAX,K,L)
OMEGA(JMAX1,K,L) = OMEGA(JMAX,K,L)
922 CONTINUE
925 CONTINUE
IF SIDBDY.NE. FREE OR WALL, THEN A DIRICHLET BOUNDARY C
IS ASSUMED AND NO MODIFICATION OF JMAX1 IS MADE.
C... FINISHED BOUNDARY CONDITIONS ON RHO, EPS, A, JN, AND OMEGA.

RETURN
END

SUBROUTINE VEL
C
C CALCULATES U, W, AND VT ON FACES FOR S, T, A, AND RHO CELL CENTE
C
PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64)
C
REAL JN
C* JN AND OMEGA ARE EXTRA USELESS ARRAYS IN THIS SUBROUTINE.
COMMON /EOM/ S(JMAX2,KMAX2,LMAX),T(JMAX2,KMAX2,LMAX),
1 A(JMAX2,KMAX2,LMAX),U(JMAX2,KMAX2,LMAX),
2 W(JMAX2,KMAX2,LMAX),JN(JMAX2,KMAX2,LMAX),
3 OMEGA(JMAX2,KMAX2,LMAX),VT(JMAX2,KMAX2,LMAX)
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RH0(JMAX2,KMAX2,LMAX)
COMMON /GEX/ GEMS,GEMR,GEMZ(JMAX1),V0LMI(JMAX1),DT(JMAX1),
1 RHFI(JMAX1),RHFI2(JMAX1),DR,DZ,DELTHI,RHFI3(JMAX1)
C
C------------------CONDITIONAL VECTOR MERGE-----------------------
LOGICAL ILL,ILL2
C ON THE FPS-264 AFIX IS WRITTEN AS:
AFIX(ILL)=FLOAT(AND(SHIFT(ILL,-31),1))
C AN ALTERNATE WAY TO WRITE AFIX ON THE FPS-264 IS:
AFIX(ILL)=FLOAT(SHIFT(ILL,-63))
C THIS SAVES SOME TIME BUT ENGINEERING IS CHECKING OUT IF THIS
C WILL ALWAYS WORK. FOR LOGICAL .TRUE. THE FPS PUTS A 1 IN BIT 32
C 11. FOR .FALSE. THE FPS PUT A 0 IN BIT 32 AND 11. THE OTHER BI
C IN A LOGICAL ARE IGNORED.
C
C ON A VAX USING VMS 4.5 AFIX CAN BE WRITTEN AS:
AFIX(ILL)=ABS(FLOAT(ILL))
C THE VAX MAKES .TRUE. -1 AND .FALSE. 0.
C
C ON AN IBM AFIX MUST BE MADE A FUNCTION BECAUSE OF THE SILLY WAY
THE IBM COMPILER CHECKS VARIABLE TYPING AT COMPILE TIME.

FUNCTION AFIX(III)
AFIX=FLOAT(III)
RETURN
END

ON THE IBM OF COURSE THE FUNCTION AFIX(III) CAN NOT BE PLACED HE
AT LEAST THAT IS THE WAY VSFORT WAS ON 3-9-87.

ON ANY CRAY MACHINE CVMGM, CVMGP, CVMGZ, CVMGN, AND CVMGT ARE AL
DEFINED WITHIN THE FORTRAN LANGUAGE AS EXTENSIONS.

CONDITIONAL VECTOR MERGE FUNCTIONS DEFINED.-----------
CVMGT(F,B,ILL)=AFIX(ILL)*F+AFIX(.NOT. ILL)*B
CVMGT2(F,B,ILL)=AFIX(ILL)*F+AFIX(.NOT. ILL)*B
CVMZPI(B)=CVMGT(0.0,1.0,B .LE. 1.0E-100)/
1 CVMGT2(1.0,B,B .LE. 1.0E-100)

CALCULATE THE VELOCITIES FROM THE MOMENTUM DENSITIES-----
FOR L=1---------------------

L=1
IM=ILMAX
DO 771 K=2,KMAX1
KM=K-1
DO 770 J=2,JMAX1
JM=J-1
U(J,K,L)=0.5*(S(J,K,L)*CVMZPI(RHO(J,K,L))=
1 S(JM,K,L)*CVMZPI(RHO(JM,K,L)))+
W(J,K,L)=0.5*(T(J,K,L)*CVMZPI(RHO(J,K,L))=
1 T(J,KM,L)*CVMZPI(RHO(J,KM,L)))+
WHAT THE BELOW VT DOES
VT(J,K,L)=0.5*RHFI(J)*(A(J,K,L)/RHO(J,K,L)+A(J,K,LM)/RHO(J,K,

770 CONTINUE
771 CONTINUE

FOR L=2,ILMAX-------------------

DO 752 L=2,ILMAX
IM=L-1
DO 751 K=2,KMAX1
KM=K-1
DO 750 J=2,JMAX1
JM=J-1
U(J,K,L)=0.5*(S(J,K,L)*CVMZPI(RHO(J,K,L))=
1 S(JM,K,L)*CVMZPI(RHO(JM,K,L)))+
W(J,K,L)=0.5*(T(J,K,L)*CVMZPI(RHO(J,K,L))=
1 T(J,KM,L)*CVMZPI(RHO(J,KM,L)))+
WHAT THE BELOW VT DOES
VT(J,K,L)=0.5*RHFI(J)*(A(J,K,L)/RHO(J,K,L)+A(J,K,LM)/RHO(J,K,

750 CONTINUE
751 CONTINUE
752 CONTINUE
C***SETTING ON AXIS RADIAL VELOCITY TO ZERO
    DO 801 L=1,LMAX
    DO 800 K=1,KMAX1
       U(1,K,L)=U(2,K,L)
   800 CONTINUE
   801 CONTINUE
C***THIS CASE IS GOOD ONLY FOR EQUATORIAL SYMMETRY
    DO 951 L=1,LMAX
    DO 950 J=1,JMAX1
       W(J,2,L)=0.0
       W(J,1,L)=0.0
   950 CONTINUE
   951 CONTINUE
C
C ON 4/13/86 AT LSU, ADDED SUBROUTINE 'VLIMIT' TO HANDLE
C THE BUSINESS OF LIMITING ALLOWED MAXIMUM VELOCITIES.
C
CALL VLIMIT
RETURN
END

SUBROUTINE VRPOW(VI, ISVI, REXP, RFCT, VO, ISVO, N, RMVAL, IMIDX, IER
    INTEGER    ISVI, ISVO, IMIDX, N, IERR
    REAL       VI(1), VO(1), RMVAL, REXP, RFCT
    REAL       ZERO, ONE, RMAXP
C
VOST(Y) = RFCT * VI(X)**REXP
  = RFCT * EXP(REXP*NLOG(VI(X))) IF VI(X) > 0
  = 0 IF VI(X) = 0
  = UNDEFINED IF VI(X) < 0
C
VI(N*ISVI) .... REAL INPUT VECTOR ........ OF LENGTH N*ISVI
ISVI .......... INTEGER INPUT SCALAR ...... STEP FOR VECTOR V
REXP .......... REAL INPUT SCALAR ........ EXPONENT
RFCT .......... REAL INPUT SCALAR ........ FACTOR
VO(N*ISVO) .... REAL OUTPUT VECTOR ........ OF LENGTH N*ISVO
ISVO .......... INTEGER OUTPUT SCALAR .... STEP FOR VECTOR V
N ............ INTEGER INPUT SCALAR .... COUNT FOR VECTORS
RMVAL .......... REAL OUTPUT SCALAR ........ MINIMUM VALUE IF
IMIDX .......... INTEGER OUTPUT SCALAR .... INDEX OF RMVAL (=
IERR .......... INTEGER OUTPUT SCALAR .... = 0 IF RMVAL >=
               = 1 IF RMVAL <
C
C
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DATA ZEROA / 1.000E-100 /  
DATA ONE  / 1.0 /  
DATA RMAXP / 8.988E+307 /  
C  
    IERR = 0  
C  
    CALL MINV(VI,ISVI,RMVAL,IMIDX,N)  
    IF (RMVAL.GE.ZERO) THEN  
        CALL VCLIP(VI,ISVO,ZEROA,RMAXP,VO,ISVO,N)  
        CALL VALG(VO,ISVI,VO,ISVO,N)  
        CALL VSMUL(VO,ISVO,REXP,VO,ISVO,N)  
        CALL VEXP(VO,ISVO,VO,ISVO,N)  
        IF (RFCT.NE.ONE) THEN  
            CALL VSMUL(VO,ISVO,RFCT,VO,ISVO,N)  
        ENDIF  
    ELSE  
        IERR = 1  
    ENDIF  
RETURN  
END
SUBROUTINE SETBDY(NCALL, ISYM)
C* AT PRESET THESE ROUTINES WHICH DEAL WITH THE POTENTIAL SOLVER
C* ALLOW ONLY PI SYMMETRY.
C
C... THIS ROUTINE SIMPLY INITIALIZES VARIOUS ARRAYS BEFORE A CALL
C BDYGEN IS MADE.
C IF NCALL = 0, THEN INNITIALIZE ALL ARRAYS IN COMMON BLOCK /B
C .GT.0, GRID HAS MOVED, SO RE-EVALUATE RBDY, JPOS, A

DIMENSION COSM AND SINM (LMAX,10).
THE DIMENSION OF RBDY, JPOS, KPOS, AND IARAY DEPENDS ON SYMMET
USED (SEE ALSO LAST DIMENSION OF BDYTRM IN ROUTINE BDYGEN):
IF ABS(ISYM) = 1 OR 8, DIMENSION THEM (2*JMAX + KMAX -1).
= 2,3, OR 9, DIMENSION THEM (JMAX + KMAX -1).

PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2,
1 KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2,
2 LMAX=64, JKM1 = JMAX+KMAX-1)

COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAV
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX2),
1 R0F3N, Z0F3N, AINEWR, AINEWZ
COMMON /BDY/ COSM(LMAX,10),SINM(LMAX,10),
1 RBDY(JKM1),JPOS(JKM1),KPOS(JKM1),JKMAX,BDYC
DIMENSION IARAY(JKM1),RB2(JKM1)

ISYMA = IABS(ISYM)
IF(NCALL.NE.0)GO TO 200
BDYCHK = 1.0
DO 5 I=1,10
5  COSM(L,I)=0.0
5  SINM(L,I)=0.0
DO 15 L=1,LMAX
PSI = FLOAT(L-1)*DTHETA + 0.5*DTHETA
15  CONTINUE

DO 15 M=1,10
PRODCT = PSI*FLOAT(M)
COSM(L,M)=COS(PRODCT)
SINM(L,M)=SIN(PRODCT)
15  CONTINUE

JKSTOP=JMAX + KMAX - 1
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)JKSTOP=2*JMAX+KMAX-1
DO 205 JK=1,JKSTOP
RBDY(JK)=0.0
JPOS(JK)=0
205  KPOS(JK)=0
I=0
K=KMAX1

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
ZK2=ZHF(K)**2
DO 215 J=2,JMAX1
  I=I+1
  RBDY(I)=SQRT(ZK2+RHF(J)**2)
215 IARAY(I)=I
IF(ISYMA.NE.1.AND.ISYMA.NE.8)GO TO 228
  K=1
  ZK2=ZHF(1)**2
  DO 225 J=2,JMAX1
    I=I+1
    RBDY(I)=SQRT(ZK2+RHF(J)**2)
225 IARAY(I)=I
228 CONTINUE
  J=JMAX1
  RJ2=RHF(J)**2
  DO 230 K=2,KMAX
    I=I+1
    RBDY(I)=SQRT(RJ2+ZHF(K)**2)
230 IARAY(I)=I
  JKMAX=I

C
C
DO 232 JK=1,JKMAX
  RB2(JK)=RBDY(JK)
CALL SORT(RB2,IARAY,JKMAX)
  DO 234 JK=1,JKMAX
234 RBDY(JK)=RB2(JK)

C
C
JKSKIP = JMAX
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)JKSKIP=2*JMAX
DO 245 I=1,JKMAX
  II=IARAY(I)
  IF(II.GT.JMAX)GO TO 238
  KPOS(I)=KMAX1
  JPOS(I)=II+1
  GO TO 245
238 IF(II.GT.JKSKIP)GO TO 240
  KPOS(I)=1
  JPOS(I)=(II-JMAX) + 1
  GO TO 245
240 JPOS(I)=JKMAX1
  KPOS(I)=(II-JKSKIP) + 1
245 CONTINUE
CONTINUE
RETURN
END

SUBROUTINE SORT(KEY,INDX,NUM)
C.. .  DOUBLE SHELL SORT FROM H.M.MURPHY 1967
C... J.E.TOHLINE GOT THIS FROM S.H.HODSON 10/17/80.
REAL KEY(1)
INTEGER INDX(1),TI
IF(NUM.LT.2)RETURN
  I=1
20 I=I+1
IF(I.LE.NUM)GO TO 20
M=I-1
30 M=M/2
   IF(M.LT.1)RETURN
   K=NUM-M
   DO 50 J=1,K
   I=J
40 IM=I+M
   IF(KEY(I).GE.KEY(IM))GO TO 50
   TK=KEY(I)
   TI=INDX(I)
   KEY(I)=KEY(IM)
   INDX(I)=INDX(IM)
   KEY(IM)=TK
   INDX(IM)=TI
   I=I-M
   IF(I.GE.1)GO TO 40
50 CONTINUE
   GO TO 30
END

SUBROUTINE BDYGEN(MAXTRM, ISYM, REDGE)
C
C .. CALL PARAMETERS ...
C MAXTRM = MAXIMUM L TO BE USED IN SPHERICAL HARMONIC EXPANSION.
C FOR GREATEST EFFICIENCY, USE 4, 8, OR 10.
C 10 IS MAXIMUM ALLOWABLE VALUE.
C ISYM = NEGATIVE, ALL MASS TOTALLY WITHIN GRID BOUNDARY R"S.
C = POSITIVE, USE GENERAL EXPANSION SINCE SOME MASS OUTSIDE.
C ABS(ISYM) = 1, NO SYMMETRIES.
C = 2, SYM. THRU EQUATORIAL PLANE, FULL 2-PI.
C = 3, PI-SYMMETRY AND SYM. THRU EQUATORIAL PLANE.
C = 8, 2-D WITH NO SYMMETRIES.
C = 9, 2-D WITH SYM. THRU EQUATORIAL PLANE.
C REDGE = 0.0, MASS CAN BE ANYWHERE IN GRID.
C .GT.0.0, MASS ENTIRELY WITHIN SPHERE OF RADIUS = REDGE.
C
C PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2, 1
KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2, 2
LMAX=64, JKM1 = JMAX+KMAX-1)
C
C COMMON /BLOK6/ DTHETA,COSIGN(JMAX),SIGN(LMAX),PI,GRAV
C* COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
C* COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX1),
C* 1 ROF3N,ZOF3N,ALNEWR,ALNEWZ
C* 1 G(JMAX2),H(KMAX2),ROF3N,ZOF3N,ALNEWR,ALNEWZ
C* 1 COMMON /OLD/ RO(JMAX2),ZO(KMAX2),RHFO(JMAX1),ZHFO(KMAX1)
C* 1 IGRID
C* DIMENSION RO(JMAX2),ZO(KMAX2),RHFO(JMAX1),ZHFO(KMAX1)
C** ALL THIS OLD GRID STUFF CAN PROBABILY BE DONE AWAY WITH HERE.
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /INSIDE/ TMASS,ENEW,ELOST,EDIF,PHICHK,KLOCAT
COMMON /BDY/ COSM(LMAX,10),SINM(LMAX,10),
             RBDY(JKM1),JPOS(JKM1),KPOS(JKM1),JMAX,BDYC

COMMON/TERMS/T00,T10,T11,T20,T21,T22, T30,T31,T32,T33, T40,T
  1 T42,T43,T44, T50,T51,T52,T53,T54,T55, T60,T61,T62,T63,T64,T
  2 , T70,T71,T72,T73,T74,T75,T76,T77, T80,T81,T82,T83,T84,T85, 
  3 T87,T88, T90,T91,T92,T93,T94,T95,T96,T97,T98,T99, T100,T101
  4 T102,T103,T104,T105,T106,T107,T108,T109,T110

C... THE DIMENSIONS OF TERM,Q,CM,SM,TRMTOT,TRMIN AND TRMOUT
C SHOULD NEVER BE ALTERED.
C THEY ALLOW FOR EXPANSIONS THROUGH L=10, M=10.

C... THE LAST DIMENSION OF ARRAY BDYTRM, HOWEVER, SHOULD BE SET E
C TO THE SIZE OF ARRAYS RBDY, JPOS, AND KPOS. (SINCE BDYTRM I
C LARGE, IT CAN, IF NEED BE, BE REDUCED FURTHER IF "MAXTRM.LT.
C THE FIRST DIMENSION OF BDYTRM MUST BE .GE. "2*NUMTRM", WHERE
C NUMTRM IS DETERMINED IN LOOP 5 OF THIS SUBROUTINE.)
DIMENSION TERM(66),Q(10),CM(10),SM(10)
DIMENSION TRMTOT(132),TRMIN(132),TRMOUT(132),BDYTRM(132,2,JK
EQUIVALENCE (TERM(1),T00)

IN A SPHERICAL HARMONIC EXPANSION, TRADITIONALLY,
C (1)   YLM(THETA,PSI) = SQRT((2L+1)/4PI*FACT0RIAL(L-M)/FACTORIA
          *PLM(X)*EXP(I*M*PSI),
          YL,-M(THETA,PSI) = (-1)**M*COMPLEX CONJUGATE(YLM),
          WHERE: X=COS(THETA) AND I=SQRT(-1).
THE EXPANSION OF BOUNDARY POTENTIALS IN TERMS OF SPHERICAL
HARMONICS LEADS TO PRODUCTS OF YLM AND COMPLEX CONJUGATE(YL
A MORE USEFUL DEFINITION OF YLM IS
C (2)   YLM(THETA,PSI) = SQRT((2L+1)/4PI)*SQRT(DLM)*BLM(THETA,PS
          *EXP(I*M*PSI).
IN THIS EXPRESSION FOR YLM, PLM(X) HAS BEEN FACTORED INTO A
NUMERICAL COEFFICIENT "COEF" AND A THETA-DEPENDENT EXPRESSI
THEN,
          DLM = (FACT0RIAL(L-M)/FACT0RIAL(L+M))*COEF**2.
IN PRACTICE, DLM ALWAYS CONSISTS OF AN ODD INTEGRAL DIVIDED
The power $n$ varies with $l$ and $m$, but for terms through $l=10$ varies from 0 to 18. The following data statement contains values of $2^{-n}$ for $n = 5$ through 18; e.g., $T_{-8} = 2^{-8}$. These terms will be used to calculate appropriate DLM's.

\[
\begin{align*}
\text{DATA } & TW5, TW6, TW7, TW8, TW9, TW10, TW11, TW12, TW13, TW14, TW15, TW16, \\
& TW17/0.03125, 0.015625, 0.0078125, 0.00390625, 0.001953125, \\
& 0.0009765625, 0.00048828125, 0.000244140625, 0.0001220703125, \\
& 0.00006103515625, 0.00003051757813, 0.00001525878907, 0.000007629394535, \\
& 0.000003814697266/ \\
\end{align*}
\]

The following statement functions are the BLM's used in the definition of $Y_l m$ in equation (2), above. Variables that are multiplied by numerical coefficients are always even powers of $\cos(\theta)$; a variable preceding a parenthetical expression to the first power; a variable trailing a parenthetical expression is always some power (either even or odd) of $\sin(\theta)$.

\[
\begin{align*}
B_{20}(A) & = 3.0*A - 1.0 \\
B_{30}(A,D) & = D*(5.0*A - 3.0) \\
B_{31}(A,D) & = (5.0*A - 1.0)*D \\
B_{40}(A,D) & = 35.0*A - 30.0*D + 3.0 \\
B_{41}(A,D,E) & = D*(7.0*A - 3.0)*E \\
B_{42}(A,D) & = (7.0*A - 1.0)*D \\
B_{50}(A,D,E) & = E*(63.0*A - 70.0*D + 15.0) \\
B_{51}(A,D,E) & = (21.0*A - 14.0*D + 1.0)*E \\
B_{52}(A,D,E) & = D*(3.0*A - 1.0)*E \\
B_{53}(A,D) & = (9.0*A - 1.0)*D \\
B_{60}(A,D,E) & = 231.0*A - 315.0*D + 105.0*E - 5.0 \\
B_{61}(A,D,E,F) & = E*(33.0*A - 30.0*D + 5.0)*F \\
B_{62}(A,D,E) & = (33.0*A - 18.0*D + 1.0)*E \\
B_{63}(A,D,E) & = D*(11.0*A - 3.0)*E \\
B_{64}(A,D) & = (11.0*A - 1.0)*D \\
B_{70}(A,D,E,F) & = F*(429.0*A - 693.0*D + 315.0*E - 35.0) \\
B_{71}(A,D,E,F) & = (429.0*A - 495.0*D + 135.0*E - 5.0)*F \\
B_{72}(A,D,E,F) & = E*(143.0*A - 110.0*D + 15.0)*F \\
B_{73}(A,D,E) & = (143.0*A - 66.0*D + 3.0)*E \\
B_{74}(A,D,E) & = D*(13.0*A - 3.0)*E \\
B_{75}(A,D) & = (13.0*A - 1.0)*D \\
B_{80}(A,D,E,F) & = 6435.0*A - 12012.0*D + 6930.0*E - 1260.0*F + \\
B_{81}(A,D,E,F,GG) & = F*(715.0*A - 1001.0*D + 385.0*E - 35.0)*GG \\
B_{82}(A,D,E,F) & = (143.0*A - 143.0*D + 33.0*E - 1.0)*F \\
B_{83}(A,D,E,F) & = E*(39.0*A - 26.0*D + 3.0)*F \\
B_{84}(A,D,E) & = (65.0*A - 26.0*D + 1.0)*E \\
B_{85}(A,D,E) & = D*(5.0*A - 1.0)*E
\end{align*}
\]
FINISHED LISTING NEEDED STATEMENT FUNCTIONS.

THE POTENTIAL AT ANY POINT (RB, THETAB, PSIB) IS GIVEN BY:

\[-GR\text{AV} \sum \text{OVER ALL } L,M \text{ (} M \neq 0 \text{ ) OF} \]

\[2.0 \cos (M \times \text{PSIB}) \times DLM \times BLM(\text{THETAB})\]

\[+ (RB the L+1) \times \text{MASSIN1}(L,M) + RB \times L \times \text{MASSOUT1}(L,M)\]

\[+ 2.0 \sin (M \times \text{PSIB}) \times DLM \times BLM(\text{THETAB})\]

\[+ (RB the L+1) \times \text{MASSIN2}(L,M) + RB \times L \times \text{MASSOUT2}(L,M)\]

PLUS \[-GR\text{AV} \sum \text{OVER ALL } L,M=0 \text{ OF} \]

\[DLO \times BL0(\text{THETAB}) \times (RB the L+1) \times \text{MASSIN1}(L,0) + RB \times L \times \text{MASSOUT}\]

GIVEN THAT THE POINT (RB, THETAB, PSIB) IS AT A SPHERICAL RAD

RSHER, THE TERMS MASSIN AND MASSOUT (EACH A FUNCTION OF L

ARE INTEGRALS OVER THE MASS DISTRIBUTION INSIDE AND OUTSIDE

RESPECTIVELY, OF RSHER. LETTING \TERM(R,\text{THETA},\text{PSI}) =

\BLM(\text{THETA}) \times \text{RH0}(R,\text{THETA},\text{PSI}) \times \text{DVOLUME}(R,\text{THETA}),

\text{MASSIN1}(L,M) = \text{SUM OVER ALL } \text{THETA, PSI, R INSIDE RSHER OF}

\text{R} \times L \times \cos (M \times \text{PSI}) \times \TERM(R,\text{THETA},\text{PSI}),\]
MASSIN2(L,M) = SUM OVER ALL \theta, \psi, r INSIDE RSPHER OF
R**L*SIN(M*\psi)*TERM(R,\theta,\psi),

MASSOUT1(L,M) = SUM OVER ALL \theta, \psi, r OUTSIDE RSPHER
R**-(L+1)*COS(M*\psi)*TERM(R,\theta,\psi),

MASSOUT2(L,M) = SUM OVER ALL \theta, \psi, r OUTSIDE RSPHER
R**-(L+1)*SIN(M*\psi)*TERM(R,\theta,\psi).

NOW BEGIN PROGRAM.

TMASS=0.0
IF(BDYCHK.NE.1.0)GO TO 990
IF(IGRID.GT.0)CALL SETBDY(IGRID,ISYM)
ISYMA = IABS(ISYM)
NTRM = 1
DO 5 LL = 1,MAXTRM
   NTRM = NTRM + LL + 1
   NUMTRM = 2*NTRM
   LELMAX = MAXTRM + 1
   JMX = JKMAX
   ISMAX = 2
   NUMOUT = NUMTRM
   IF(ISYM.GT.0) GO TO 6
   JMX = 1
   ISMAX = 1
   NUMOUT = 1
5 CONTINUE
DO 8 I = 1,NUMTRM
   TRMIN(I) = 0.0
8 DO 9 I = 1,NUMOUT
   TRMOUT(I) = 0.0
9 DO 10 JK = 1,JMX
     DO 10 IS = 1,ISMAX
        DO 10 I = 1,NUMTRM
           BDYTRM(I,IS,JK) =0.0
10 F1 = 1.0
   IF(ISYMA.EQ.2.OR.ISYMA.EQ.9)F1=0.0
   IF(ISYMA.EQ.3)F1=0.0
   F2 = 1.0
   IF(ISYMA.EQ.2.OR.ISYMA.EQ.9)F2=2.0
   IF(ISYMA.EQ.3)F2=4.0

BEGIN INTEGRALS OVER MASS DISTRIBUTION. ...
DO 300 K = 2,KMAX
KP = K+1
ZZ = ZHF(K)
Z2 = ZZ*ZZ
ZDEL = Z(KP) - Z(K)
DO 300 J = 2,JMAX
JP = J+1
RR = RHF(J)
R2 = RR*RR
RSPHER = SQRT(R2 + Z2)
IF(REDGE.GT.0.0.AND.RSPHER.GT.REDGE)GO TO 300

C
C STEP ONE: FOR THIS R,THETA, CALCULATE THE PRODUCT
C BLM(THETA)*R**L FOR ALL L,M. THIS PRODUCT WILL E
C ARRAY TERM, SINCE TERM IS EQUIVALENCED TO T00, T1
C
C = ZZ/RSPHER
S = RR/RSPHER
C2 = C*C
C4 = C2*C2
S2 = S*S
S4 = S2*S2
IF(MAXTRM.LE.4)GO TO 602
C6 = C2*C4
C8 = C4*C4
S5 = S*S4
S6 = S2*S4
S7 = S*S6
S8 = S4*S4
IF(MAXTRM.LE.8)GO TO 602
C10 = C2*C8
S9 = S*S8
S10 = S2*S8
602 CONTINUE
DO 604 LL = 1,10
Q(LL) = 0.0
DO 605 I = 1,66
TERM(I) = 0.0
DO 608 LL = 1,10
Q(LL) = RSPHER**LL
608 CONTINUE
T00 = 1.0
T01 = C*Q(1)*F1
T20 = B20(C2)*Q(2)
T30 = B30(C2,C)*Q(3)
T40 = B40(C4,C2) *Q(4)
IF(ISYMA.GE.8) GO TO 611
T22 = S2*Q(2)
T42 = B42(C2,S2) *Q(4)
T44 = S4*Q(4)
IF(ISYMA.EQ.3) GO TO 611
T11 = S*Q(1)
T31 = B31(C2,S) *Q(3)
T33 = S3*Q(3)
IF(ISYMA.EQ.2) GO TO 611
T21 = C*S*Q(2)
T32 = C*S2*Q(3)
T41 = B41(C2,C,S) *Q(4)
T43 = C*S3*Q(4)
IF(MAXTRM.LE.4) GO TO 620

611 IF(MAXTRM.LE.4) GO TO 620
611  IF(ISYMA.GE.8) GO TO 613
C...
T50 = B50(C4,C2,C) *Q(5)*F1
T60 = B60(C6,C4,C2) *Q(6)
T70 = B70(C6,C4,C2,C) *Q(7)*F1
T80 = B80(C8,C6,C4,C2) *Q(8)
IF(ISYMA.GE.8) GO TO 613
OH = Q(6)
T62 = B62(C4,C2,S2) *QH
T64 = B64(C2,S4) *QH
T66 = S6*QH
OH = Q(8)
T82 = B82(C6,C4,C2,S2) *QH
T84 = B84(C4,C2,S4) *QH
T86 = B86(C2,S6) *QH
T88 = S8*QH
IF(ISYMA.EQ.3) GO TO 613
OH = Q(5)
T51 = B51(C4,C2,S) *QH
T53 = B53(C2,S3) *QH
T55 = S5*QH
OH = Q(7)
T71 = B71(C6,C4,C2,S) *QH
T73 = B73(C4,C2,S3) *QH
T75 = B75(C2,S5) *QH
T77 = S7*QH
IF(ISYMA.EQ.2) GO TO 613
T52 = B52(C2,C,S2) *Q(5)
T54 = C*S4*Q(5)
OH = Q(6)
T61 = B61(C4,C2,C,S) *QH
T63 = B63(C2,C,S3) *QH
T65 = C*S5*QH
OH = Q(7)
T72 = B72(C4,C2,C,S2) *QH
T74 = B74(C2,C,S4) *QH
T76 = C*S6*QH
OH = Q(8)
T81 = B81(C6,C4,C2,C,S) *QH
T83 = B83(C4,C2,C,S3) *QH
T85 = B85(C2,C,S5)*QH
T87 = C*S7*QH

613 IF(MAXTRM.LE.8)GO TO 620
613  IF(ISYMA.GE.8) GO TO 617
C...
T90 = B90(C8,C6,C4,C2,C) *Q(9)*F1
T100 = B100(C10,C8,C6,C4,C2) *Q(10)
IF(ISYMA.GE.8)GO TO 617
QH = Q(10)
T102 = B102(C8,C6,C4,C2,S2) *QH
T104 = B104(C6,C4,C2,S4) *QH
T106 = B106(C4,C2,S6) *QH
T108 = B108(C2,S8) *QH
T1010 = S10*QH
IF(ISYMA.EQ.3) GO TO 617
QH = Q(9)
T91 = B91(C8,C6,C4,C2,S)*QH
T93 = B93(C6,C4,C2,S3)*QH
T95 = B95(C4,C2,S5) *QH
T97 = B97(C2,S7) *QH
T99 = S9*QH
IF(ISYMA.EQ.2) GO TO 617
T92 = B92(C6,C4,C2,C,S2) *QH
T94 = B94(C4,C2,C,S4) *QH
T96 = B96(C2,C,S6) *QH
T98 = C*S8*QH
QH = Q(10)
T101 = B101(C8,C6,C4,C2,C,S) *QH
T103 = B103(C6,C4,C2,C,S3) *QH
T105 = B105(C4,C2,C,S5) *QH
T107 = B107(C2,C,S7) *QH
T109 = C*S9*QH
617 CONTINUE
620 CONTINUE

C
C
C STEP TWO: CALCULATE MASSIN1 AND MASSIN2. THESE MASSIN TER
C DEPEND ON L AND M, AND IN GENERAL ON THE SPHERIC
C OF THE BOUNDARY ZONE BEING CONSIDERED. FOR EACH
C TERM (I=1,NUMTRM,2), AND FOR EACH BOUNDARY CELL
C (JK=1,JKMAX), MASSIN1 IS STORED IN BDYTRM(I,1,JK
C AND MASSIN2 IS STORED IN BDYTRM(I+1,1,JK).
C
C
C... DEPENDING ON CHOSEN SYMMETRY, F2 = 1.0,2.0 OR 4.0 TO AC
C FOR ALL MASS.
DV = 0.5*DTHETA*ZDEL*(R(JP)**2 - R(J)**2)*F2
DO 204 M = 1,10
CM(M) = 0.0
SM(M) = 0.0
SUMASS = 0.0
DO 210 L = 1,LMAX
C...
C H = MASS IN SINGLE GRID CELL (DV HAS MASS SYMMETRIES BU
H = RHO(J,K,L)*DV
SUMASS = SUMASS + H
DO 210 M = 1,MAXTRM
CM(M) = CM(M) + COSM(L,M)*H
SM(M) = SM(M) + SINM(L,M)*H
210 CONTINUE
DO 213 I = 1,NUMTRM
213 TRMTOT(I) = 0.0

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
NCNT = 0
DO 225 LEL = 1,LELMAX
LL = LEL - 1
MMAX = LEL
DO 225 MM = 1,MMAX
M = MM - 1
NCNT = NCNT + 1
IP = 2*NCNT
I = IP - 1
IF(M.EQ.0) GO TO 216
TRMTOT(I) = TERM(NCNT)*CM(M)
TRMTOT(IP) = TERM(NCNT)*SM(M)
GO TO 225
216 TRMTOT(I) = TERM(NCNT)*SUMASS
225 CONTINUE
TMASS = TMASS + TRMTOT(1)
C STORE SUM OF TERMS, TO BE USED LATER FOR EACH BNDRY CELL.
IF(ISYM.LT.0) GO TO 1250
C RSPHER**L HAS BEEN USED IN EXPANSIONS.
ICHK = 0
DO 235 JK = 1,JKMAX
IF(RSPHER.GT.RBDY(JK)) GO TO 230
C MASS RING IS INTERIOR TO RBDY(JK).
DO 228 I = 1,NUMTRM
BDYTRM(I,1,JK) = BDYTRM(I,1,JK) + TRMTOT(I)
GO TO 235
230 ICHK = ICHK + 1
235 CONTINUE
C TERMS FOR SAME RING OF MASS SHOULD BE DONE AGAIN IF ITS
C LIES OUTSIDE SOME BOUNDARY RADII.
IF(ICHK.EQ.0) GO TO 300
C
C STEP THREE: CALCULATE MASSOUT1 AND MASSOUT2. FOR EACH L,M
C (OPTIONAL) (I=1,NUMTRM,2) AND FOR EACH BOUNDARY CELL (JK=
C MASSOUT1 IS STORED IN BDYTRM(I,2,JK) AND MASSO
C STORED IN BDYTRM(I+1,2,JK).
C NOTE: FOR ANY PARTICULAR RING OF MASS AT S
C RADIUS R, ITS CONTRIBUTION TO MASSOUT IS JUST
C R**-(2L+1) TIMES ITS CONTRIBUTION TO MASSIN.
C
C TRMTOT(1) = TRMTOT(1)/RSPHER
NCNT = 2
DO 240 LL = 1,MAXTRM
IPWR = -(2*LL + 1)
MMAX = 2*(LL + 1)
QH = RSPHER**IPWR
DO 240 MM = 1,MMAX
NCNT = NCNT + 1
TRMTOT(NCNT) = TRMTOT(NCNT)*QH
240 CONTINUE
C RSPHER**-(L+1) HAS BEEN USED IN EXPANSIONS.
DO 245 JK = 1,JKMAX
IF(RSPHER.LE.RBDY(JK)) GO TO 245

C... MASS RING IS OUTSIDE RBDY(JK).
DO 245 I = 1,NUMTRM
BDYTRM(I,2,JK) = BDYTRM(I,2,JK) + TRMTOT(I)
245 CONTINUE

GO TO 300
1250 CONTINUE

C... ALL MASS INSIDE GRID BNDRY R'S, SO BDYTRM SAME FOR ALL
DO 1255 I = 1,NUMTRM
BDYTRM(I,1,1) = BDYTRM(I,1,1) + TRMTOT(I)
1255 CONTINUE

300 CONTINUE

FINISHED CALCULATING MASSIN1, MASSIN2, MASSOUT1, AND MASSOU

NOW, FOR EACH BOUNDARY CELL, CALCULATE SPECIFIC VALUE OF TH

DO 400 JK = 1,JKMAX
J = JPOS(JK)
K = KPOS(JK)
RSPHER = RBDY(JK)

STEP FOUR: CALCULATE THE PRODUCT DLM*BLC AND STORE RESULTS
ARRAY TERM.

C = ZHF(K)/RSPHER
S = RHF(J)/RSPHER
C2 = C*C
C4 = C2*C2
S2 = S*S
S3 = S*S2
S4 = S2*S2
IF(MAXTRM.LE.4) GO TO 304
C6 = C2*C4
C8 = C4*C4
S5 = S*S4
S6 = S2*S4
S7 = S*S6
S8 = S4*S4
IF(MAXTRM.LE.8) GO TO 304
C10 = C2*C8
S9 = S*S8
S10 = S2*S8

304 CONTINUE
DO 305 I = 1,66
TERM(I) = 0.0
305 CONTINUE

C... EL.EQ.0 THRU 4
T00 = 1.0
T10 = C*F1
T20 = 0.25*B20(C2)
T30 = 0.25*B30(C2,C)*F1
T40 = TW6 *B40(C4,C2)
IF(ISYMA.GE.8)GO TO 307
T22 = 0.375*S2
T42 = 5.0*TW5 *B42(C2,S2)
T44 = 35.0*TW7 *S4
IF(ISYMA.EQ.3)GO TO 307
T11 = 0.5*S
T31 = 0.1875*B31(C2,C2)
T33 = 0.3125*53
IF(ISYMA.EQ.2) GO TO 307
T21 = 1.5*C*S
T32 = 1.875*C*S2
T41 = 0.3125 *B41(C2,C,5)
T43 = 2.1875 *0*53
IF(MAXTRM.LE.4)GO TO 310

307 IF(MAXTRM.LE.8)GO TO 310
C...

T50 = TW6 *B50(C4,C2,C2,C) *F1
T60 = TW8 *B60(C6,C4,C2)
T70 = TW8 *B70(C6,C4,C2,C) *F1
T80 = TW14 *B80(C8,C6,C4,C2)
IF(ISYMA.GE.8)GO TO 308
T62 = 105.0*TW10 *B62(C4,C2,S2)
T64 = 63.0*TW9 *B64(C2,S4)
T66 = 231.0*TW10 *S6
T82 = 315.0*TW12 *B82(C6,C4,C2,S2)
T84 = 693.0*TW13 *B84(C4,C2,S4)
T86 = 429.0*TW12 *B86(C2,S6)
T88 = 6435.0*TW15 *S8
IF(ISYMA.EQ.3)GO TO 308

308 IF(MAXTRM.LE.8)GO TO 310
C...

T51 = 15.0*TW7 *B51(C4,C2,S)
T53 = 35.0*TW8 *B53(C2,S3)
T55 = 63.0*TW8 *S5
T71 = 7.0*TW11 *B71(C6,C4,C2,S)
T73 = 21.0*TW11 *B73(C4,C2,S3)
T75 = 231.0*TW11 *B75(C2,S5)
T77 = 429.0*TW11 *S7
IF(ISYMA.EQ.2)GO TO 308

T52 = 105.0*TW5*B52(C2,C,S2)
T54 = 315.0*TW7 *C*S4
T61 = 21.0*TW7 *B61(C4,C2,C,S)
T63 = 105.0*TW8 *B63(C2,C,S3)
T65 = 693.0*TW8*C*S5
T72 = 21.0*TW10 *B72(C4,C2,C,S2)
T74 = 231.0*TW9 *B74(C2,C,S4)
T76 = 3003.0*TW10 *C*S6
T81 = 9.0*TW11 *B81(C6,C4,C2,C,S)
T83 = 1155.0*TW11 *B83(C4,C2,C,S3)
T85 = 9009.0*TW11 *B85(C2,C,S5)
T87 = 6435.0*TW11 *C*S7

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
T90 = TW14 *B90(C8,C6,C4,C2,C)*F1
T100 = TW16 *B100(C10,C8,C6,C4,C2)
IF(ISYMA.GE.8)GO TO 309
T102 = 165.0*TW17 *B102(C8,C6,C4,C2,S2)
T104 = 2145.0*TW15 *B104(C6,C4,C2,S4)
T106 = 2145.0*TW18 *B106(C4,C2,S6)
T108 = 12155.0*TW17 *B108(C2,S8)
T1010 = 46189.0*TW18 *S10
IF(ISYMA.EQ.3) GO TO 309
T91 = 45.0*TW15 *B91(C8,C6,C4,C2,S)
T93 = 1155.0*TW14 *B93(C6,C4,C2,S3)
T95 = 1287.0*TW14 *B95(C4,C2,S5)
T97 = 6435.0*TW16 *B97(C2,S7)
T99 = 12155.0*TW16*S9
IF(ISYMA.EQ.2)GO TO 309
T92 = 495.0*TW12 *B92(C6,C4,C2,C,S2)
T94 = 45045.0*TW13 *B94(C4,C2,C,S4)
T96 = 2145.0*TW12 *B96(C2,C,S6)
T98 = 109395.0*TW15 *C*S8
T101 = 55.0*TW15 *B101(C8,C6,C4,C2,C,S)
T103 = 2145.0*TW14 *B103(C6,C4,C2,C,S3)
T105 = 429.0*TW14 *B105(C4,C2,C,S5)
T107 = 36465.0*TW16 *B107(C2,C,S7)
T109 = 230945.0*TW16 *C*S9
309 CONTINUE
310 CONTINUE

STEP FIVE: COMBINE MASSIN (CALLED TRMIN HERE) AND MASSOUT TRMOUT) TERMS APPROPRIATELY, KEEPING COSINE DEP TERMS (TRMTOT(I=ODD)) SEPARATE FROM SINE DEPEND (TRMTOT(I=EVEN)).

IF(ISYM.LT.0)GO TO 1321
DO 312 I = 1,NUMTRM
TRMIN(I) = BDYTRM(I,1,JK)
312 TRMOUT(I) = BDYTRM(I,2,JK)
NCNT = 0
DO 320 LEL = 1,LELMAX
LL = LEL - 1
MMAX = LEL
IF(LL.EQ.0)GO TO 314
RIN = 1.0/RSPHER**MMAX
ROUT = RSPHER**LL
GO TO 315
314 RIN = 1.0/RSPHER
ROUT = 1.0
315 CONTINUE
DO 320 MM = 1,MMAX
NCNT = NCNT + 1
IP = NCNT*2
I = IP - 1
B = TERM(NCNT)*RIN

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
H = TERM(NCNT)*ROUT
TRMTOT(I) = B*TRMIN(I) + H*TRMOUT(I)
TRMTOT(IP) = B*TRMIN(IP) + H*TRMOUT(IP)
GO TO 331

C... ASSUMING NO MASS OUTSIDE GRID RADIUS.
DO 1323 I = 1,NUMTRM
TRMIN(I) = B*TRMIN(I)
NCNT = 0
DO 1330 LEL = 1,LELMAX
LL = LEL - 1
MMAX = LEL
RIN = 1.0/RSPHER**MMAX
DO 1330 MM = 1,MMAX
NCNT = NCNT + 1
IP = 2*NCNT
I = IP - 1
B = TERM(NCNT)*RIN
TRMTOT(I) = B*TRMIN(I)
TRMTOT(IP) = B*TRMIN(IP)
CONTINUE

FINALLY, STEP SIX: SUM OVER ALL L,M TERMS, TAKING INTO ACCOUNT THE PSIB ANGLE DEPENDENCE.

DO 350 L = 1,LMAX
DO 335 M = 1,MAXTRM
CM(M) = COSM(L,M)
SM(M) = SINM(L,M)
NCNT = 0
SUM = 0.0
DO 345 LEL = 1,LELMAX
LL = LEL - 1
MMAX = LEL
DO 345 MM = 1,MMAX
M = MM - 1
NCNT = NCNT + 2
I = NCNT - 1
IP = NCNT
IF(M.EQ.0)GO TO 340
SUM = SUM + 2.0*(CM(M)*TRMTOT(I) + SM(M)*TRMTOT(IP))
GO TO 345
SUM = SUM + TRMTOT(I)
CONTINUE
340 SUM = SUM + TRMTOT(IP)
345 CONTINUE
350 PHI(J,K,L) = - GRAY*SUM
400 CONTINUE

FINISHED HARD WORK. NOW TIDY UP BOUNDARY SYMMETRIES.

IF(ISYMA.EQ.1)GO TO 550
C... FOR 2-D OR PI-SYM PROBLEMS, SET BOUNDARY CONDITIONS ON Z-AXI
IF(ISYMA.NE.3.AND.ISYMA.NE.8.AND.ISYMA.NE.9)GO TO 525
DO 505 L=1,LMAX
505 PHI(1,KMAX1,L) = PHI(2,KMAX1,L)
DO 510 L=1,LMAX
510 PHI(1,1,L) = PHI(2,1,L)
GO TO 580
C
C... FOR SYM THROUGH EQUATORIAL PLANE, SET BNDRY CONDITION THRU P
525 IF(ISYMA.NE.2.AND.ISYMA.NE.3.AND.ISYMA.NE.9)GO TO 550
DO 530 L=1,LMAX
530 PHI(JMAX1,1,L) = PHI(JMAX1,2,L)
DO 580
C
C.. . IF NO SYM THRU Z-AXIS, EQUATE CORRECT PHI'S AT J=1.
550 LHAF = LMAX/2
DO 560 L=1,LHAF
LP=L+LHAF
PHI(1,KMAX1,L) = PHI(2,KMAX1,LP)
PHI(1,KMAX1,LP) = PHI(2,KMAX1,L)
IF(ISYMA.NE.1)GO TO 560
PHI(1,1,L) = PHI(2,1,LP)
PHI(1,1,LP) = PHI(2,1,L)
560 CONTINUE
580 CONTINUE
RETURN
C
C 990 WRITE(6,101)BDYCHK
101 FORMAT(/,' STOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOP
1 5X,'BDYCHK =',IP10.2,/,5X,'SETBDY HAS NOT BEEN CALLED
2 5X,'IT MUST BE CALLED AT LEAST ONCE IN ORDER TO INITIA
3 5X,'THE ARRAYS IN COMMON BLOCK /BDY/',/,' STOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPST
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX1),
  ROF3N,ZOF3N,A1NEWR,A1NEWZ
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /COEFS/ COEF(JMAX,LMAX,2)
COMMON /SETPHI/ WSAVE(MWSAVE)
C  (ARRAY A1 IS USED IN FFT;
C    DIMENSION IT 2*LMAX FOR PI-SYM RUN.)
DIMENSION A1(LMAX2)
C  (THESE NEXT ARRAYS ARE USED IN BLKTRI.)
DIMENSION AN(IKK),BN(IKK),CN(IKK),
  AM(IJJ),BM(IJJ),CM(IJJ),Y(IJJ,IKK),WFW(MFWF)
C  (THESE NEXT ARRAYS STORE QUANTITIES USED TO FIND BLKT
C    COEFFICIENTS. ARRAY C STORES LAPLACIAN'S ANGULAR
C    OPERATOR.)
DIMENSION C(IJJ),DENOMR(JMAX),RD2(JMAX),RD3(JMAX),
  DENOMZ(KMAX),ZD2(KMAX),XLAMM(L2P1)
C
ISYMA = IABS(ISYM)
IF(ISYMA.EQ.1.OR.ISYMA.EQ.2)GO TO 990
N = 2*LMAX
NL2 = LMAX/2
IF(LMAX.EQ.1) N=1
NL = NL2 + 1
SF1=1.0
SF2=1.0/FLOAT(N)
C... ZERO ARRAY A1.
DO 20 L = 1,N
A1(L)=0.0
20 CONTINUE
C... PUT, FOR CONVENIENCE, ALL RHO'S INTO PHI ARRAY -- LEAVE BNDRY
C    ALONE.
DO 30 I=1,LMAX
DO 30 K=2,KMAX
DO 30 J=2,JMAX
PHI(J,K,L)=RHO(J,K,L)
30 CONTINUE
IF(LMAX.EQ.1) GO TO 200
C... NOW, FOR ALL J,K, CALCULATE FOURIER TRANSFORM OF RHO'S AND B
C    PHI'S.
DO 40 K=2,KMAX1
DO 40 J=2,JMAX1
C... PUT DENSITIES INTO ARRAY A1, ONE RING AT A TIME.
   DO 35 L=1,LMAX
      LP=L+LMAX
      A1(LP)=PHI(J,K,L)
   35 CONTINUE
   CALL RFFT(N,A1,WSAVE)
C... PERFORM FOURIER TRANSFORMATION
C (BE SURE WSAVE ARRAY HAS BEEN INITIALIZED IN ROUTINE SETUP.)
C...
NOW PUT COSINE COEFFICIENTS INTO PHI(L=1,N1) AND SINE
COEFFICIENTS INTO PHI(L=N1+1,LMAX). REMEMBER FOR PI-SYM TH
ONLY EVEN MODES ARE NON-ZERO.
PHI(J,K,1)=A1(1)*SF1
PHI(J,K,N1)=A1(N)*SF1
DO 37 I=2,NL2
L1=I
L2=I+NL2
IA1=4*I-4
IA2=4*I-3
PHI(J,K,L1)=A1(IA1)*SF1
PHI(J,K,L2)=A1(IA2)*SF1
37 CONTINUE

IF(K.NE.2)GO TO 1045
C...
FROM A1 ARRAY, FIGURE OUT AMPLITUDE AND PHASES OF VARIO
Modes. FOR PI-SYM, ONLY EVEN MODES ARE NON-ZERO.
X1=ABS(A1(1))
IF(X1.EQ.0.0)GO TO 1045
MODE=1
COEF(J,MODE,1)=X1
COEF(J,MODE,2)=0.0
NSTOP=N-1
DO 1034 I=4,NSTOP,4
MODE=MODE+1
IP=I+1
C AC IS COSINE COEFFICIENT
AC=A1(I)
C AS IS SINE COEFFICIENT
AS=A1(IP)
AMP=2.0*SQRT(AC**2 + AS**2)/X1
IF(AC.NE.0.0)PHILAG=ATAN(-AS/AC)
IF(AC.EQ.0.0)PHILAG=PI*0.5
IF(AC.EQ.0.0.AND.AS.EQ.0.0)PHILAG=4.0*PI
IF(AC.LT.0.0)PHILAG=PHILAG + PI
IF(PHILAG.LT.0.0)PHILAG=PHILAG + 2.0*PI
COEF(J,MODE,1)=AMP
COEF(J,MODE,2)=PHILAG*180./PI
1034 CONTINUE
IF(MODE.GE.8)GO TO 1045
COEF(J,8,1)=2.0*A1(N)/X1
COEF(J,8,2)=0.0
1045 CONTINUE
C...
FINISHED AMPLITUDES AND PHASE ANGLES.
DO 38 L=1,N
A1(L) = 0.0
38 CONTINUE
40 CONTINUE
200 CONTINUE
C...
THROUGH WITH FOURIER TRANSFORMATIONS; NOW SOLVE 2-D EQUATIO
C...
SET UP VARIOUS OPERATORS.
DTHET2=1.0/(DTHETA*DTHETA)
PIG4=4.0*PI*GRAV
RD2(1)=0.5/RHF(2)
RD3(1)=1.0/RHF(1)
DO 210 J=2,JMAX
   DENOMR(J)=1.0/(RHF(J+1)-RHF(J-1))
   RD2(J)=1.0/(RHF(J+1)-RHF(J))
   RD3(J)=1.0/RHF(J)
210 CONTINUE
ZD2(1)=0.5/ZHF(2)
DO 220 K=2,KMAX
   DENOMZ(K)=1.0/(ZHF(K+1)-ZHF(K-1))
   ZD2(K)=1.0/(ZHF(K+1)-ZHF(K))
220 CONTINUE
DO 230 J=2,JMAX
   I=J-1
   CM(I)=2.*(0.5*RD3(J)+RD2(J))*DENOMR(J)
   AM(I)=2.*(RD2(J-1)-0.5*RD3(J))*DENOMR(J)
   C(I)=RD3(J)*RD3(J)*DTHET2
230 CONTINUE
DO 240 K=2,KMAX
   I=K-1
   CN(I)=2.*ZD2(K)*DENOMZ(K)
   AN(I)=2.*ZD2(K-1)*DENOMZ(K)
   BN(I)=-CN(I)-AN(I)
240 CONTINUE
C
C   ... THE BOUNDARY CONDITIONS ON THE POISSON EQUATION ARE ALL HIDDEN
C   IN HOW YOU SPECIFY THE FIRST AND LAST VALUES OF ARRAYS CM, AM, BM, CN, AN, AND BN.
C   DIRICHLET CONDITION AT JMAX....
   IMAX=JMAX-1
   CMMAX=CM(IMAX)
   CM(IMAX)=0.0
   CMMAX IS USED IN LOOP 330 AND AT STATEMENT 331 BELOW TO COMPLETE CONDITION.
C   DIRICHLET CONDITION AT KMAX....
   IMAX=KMAX-1
   CNMAX=CN(IMAX)
   CN(IMAX)=0.0
   CNMAX IS USED IN LOOP 320 BELOW TO COMPLETE CONDITION.
C   NEUMANN CONDITION AT Z-AXIS (J=2)....
   AM(1)=0.0
   SEE MODIFICATION TO BM(1) AT STATEMENT #341 BELOW.
C   IF(ISYMA.NE.1.AND.ISYMA.NE.8)GO TO 242
C   DIRICHLET CONDITION AT K=2....
   ANMIN=AN(1)
   AN(1)=0.0
   ANMIN IS USED IN LOOP 325 BELOW TO COMPLETE CONDITION.
C   IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)GO TO 245
242 CONTINUE
NEUMANN CONDITION AT K=2 (SYM. THRU EQUATORIAL PLANE).....
BN(1)=BN(1)+AN(1)
AN(1)=0.0

CONTINUE

DO 250 M1=1,N1
M = 2*(M1-1)
EM=FLOAT(M)
XLAMM(M1)=COS(EM*DTHETA)
CONTINUE

COEFFICIENTS BM AND Y WILL VARY WITH M, SO THEY HAVEN'T BEEN
CALCULATED YET.
NOW FOR EACH VALUE OF L, THRU LMAX, CALCULATE PHI'S IN TRANS
SPACE.
DO 500 L=1,LMAX
IF(L.LE.N1) M1=L
IF(L.GT.N1) M1=L + 1 - N1
DO 310 K=2,KMAX
IK=K-1
DO 310 J=2,JMAX
IJ=J-1
Y(IJ,IK) = PIG4*PHI(J,K,L)
CONTINUE

NOW CALCULATE BM'S.
JSTOP = JMAX-2
DO 340 J=2,JSTOP
BM(J)=-CM(J)-AM(J)+2.0*(XLAMM(M1)-1.0)*C(J)
CONTINUE

MODIFY Y AND BM ACCORDING TO THE CHOSEN BOUNDARY CONDITIONS

DIRICHLET CONDITION AT JMAX.....
J=JMAX
J1=J+1
IJ=J-1
DO 330 K=2,KMAX
IK=K-1
Y(IJ,IK) = Y(IJ,IK) - CMMAX*PHI(J1,K,L)
CONTINUE

I=JMAX-1
BM(I)=-CMMAX-AM(I)+2.0*(XLAMM(M1)-1.0)*C(I)
IS THE STATEMENT NUMBER ABOVE THAT IS NOT REFERENCED.

DIRICHLET CONDITION AT KMAX.....
K=KMAX
K1=K+1
IK=K-1
DO 320 J=2,JMAX
IJ=J-1
Y(IJ,IK) = Y(IJ,IK) - CNMAX*PHI(J,K1,L)
CONTINUE

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
NEUMANN CONDITION AT Z-AXIS (J=2).....
   BM(1) = -CM(1) + 2.0*(XLAMM(M1)-1.0)*C(1)
341 IS THE STATEMENT NUMBER ABOVE THAT IS NOT REFERENCED.
IF(ISYMA.NE.1.AND.ISYMA.NE.8)GO TO 326
DIRICHLET CONDITION AT K=2.....
   IK=1
   DO 325 J=2,JMAX
      IJ=J-1
      Y(IJ,IK) = Y(IJ,IK) - ANMIN*PHI(J,1,L)
325 CONTINUE
326 CONTINUE

THIS COMPLETES THE SET UP OF COEFFICIENTS FOR GIVEN M.

DO 350 K=2,KMAX
   IK=K-1
   DO 350 J=2,JMAX
      IJ=J-1
      PHI(J,K,L)=Y(IJ,IK)
350 CONTINUE

NOW DO ANOTHER VALUE OF M.

IF(LMAX.EQ.1) GO TO 550

ALL TRANSFORMED PHI'S HAVE BEEN CALCULATED. NOW OBTAIN REAL PHI'S BY FOURIER ANALYSIS.

AL(L)=0.0

CALL RFFTB(N,AL,WSAVE)
C
C
DO 520 L=1,LMAX
520 PHI(J,K,L)=AL(L)*SF2
540 CONTINUE
550 CONTINUE
C
C
CALL ZAXPHI TO DETERMINE PHI ON Z-AXIS.
CALL ZAXPHI(10,0,ISYM)
C
C
CALCULATION OF PHI'S IS NOW FINISHED.
DO 580 L=1,LMAX
DO 570 K=1,KMAX
PHI(1,K,L)=PHI(2,K,L)
570 CONTINUE
IF(ISYMA.EQ.1.OR.ISYMA.EQ.8)GO TO 576
DO 575 J=1,JMAX2
575 PHI(J,1,L)=PHI(J,2,L)
576 CONTINUE
580 CONTINUE
RETURN
990 WRITE(6,101)ISYM
101 FORMAT(///,' STOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOP'//,
1 5X,'ISYM =',15,/,'POT3 DOES NOT YET ALLOW FULL 2PI CAL'//,
2       '2ON.',//,
4       ' STOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOPSTOP'//,
RETURN
END
SUBROUTINE ZAXPHI(NPOINT,IPRINT,ISYM)
C
PARAMETER (JMAX=64, JMAX1 = JMAX+1, JMAX2 = JMAX+2, IJ = JMA
KMAX=16, KMAX1 = KMAX+1, KMAX2 = KMAX+2, IK = KMA
2 LMAX=64, L2 = LMAX)
C (IT LOOKS LIKE L2 ONLY NEEDS TO = LMAX/2 IF DOING FULL 2*PI, C
BUT IT SHOULDN'T HURT TO LEAVE IT AS LMAX ALWAYS.)
C
COMMON /BLOK6/ DTHETA,COSIGN(LMAX),SIGN(LMAX),PI,GRAV
C* COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX1),ZHF(KMAX1),
 COMMON /GRID/ R(JMAX2),Z(KMAX2),RHF(JMAX2),ZHF(KMAX2),
1 ROF3N,ZOFR3N,A1NEWR,A1NEWZ
C* 1 G(JMAX2),H(KMAX2),ROF3N,ZOFR3N,A1NEWR,A1NEWZ
COMMON /POIS/ PHI(JMAX2,KMAX2,LMAX),RHO(JMAX2,KMAX2,LMAX)
COMMON /COEFS/ COEF(JMAX,LMAX,2)
COMMON /INSIDE/ TMASS,ENEW,ELOST,EDIF,PHICHK,KLOCAT
DIMENSION X(10),PHI2(10),PH(L2),PHAC(KMAX2)
C
COMMON /INSIDE/ TMASS,ENEW,ELOST,EDIF,PHICHK,KLOCAT
C
100 FORMAT(///,' YOU ARE LIMITED TO A 10-POINT INTERPOLATION HER'//,
1 1UBROUTINE ZAXPHI.'//,' BUT YOU PUT NPOINT =',14,///)
101 FORMAT(1P11E11.3)
C FORMAT (20X, I5, ' COEFS=', 1P8E12.3,/', 25X,' PHASE=', 0P8F12.2)
C WITH X AND PH2 DIMENSIONED 10, 10-POINT INTERPOLATION IS TH
C MAXIMUM YOU CAN DO.
C ISYMA=IABS(ISYM)
C IF(NPOINT.GT.10) WRITE(6,100) NPOINT
C IF(NPOINT.GT.10) NPOINT=10
C XPOINT=0.0
C LM=LMAX
C N=NPOINT
C ICHK=0
C ISYMA=IABS(ISYM)
C IF(MOD(N,2).NE.0) ICHK=1
C N=N-ICHK
C NUM=N/2
C Ipek=N+NUM+1
C JSP=NUM+1
C DO 10 J=2,JSP
C I1=NUM+(J-1)
C I2=NUM-(J-2)
C X(I1)=RHF(J)
C X(I2)=-RHF(J)
C 10 CONTINUE
C IF(ICHK.EQ.1) X(ISPECL)=RHF(NUM+2)
C DO 50 K=2,KMAX1
C PHMAX=0.0
C FOR GIVEN K, CALCULATE PHI ON Z-AXIS AT ALL ANGLES. STORE
C RESULTS IN PH(L) ARRAY AND IN PHI(JMAX2,K,L).
C LHAF=LMAX/2
C DO 20 L=1,LM
C LP=L
C IF(ISYMA.EQ.1.OR.ISYMA.EQ.2)LP=L+LHAF
C IF(LP.GT.LMAX)LP=L-LHAF
C DO 15 J=2,JSP
C I1=NUM+(J-1)
C I2=NUM-(J-2)
C PHI2(I1)=PHI(J,K,L)
C PHI2(I2)=PHI(J,K,LP)
C 15 CONTINUE
C IF(ICHK.EQ.1) PHI2(ISPECL)=PHI(NUM+2,K,L)
C IM=NPOINT-1
C DO 18 I=1,IM
C PI=PHI2(I)
C XINV=1.0/X(I)
C XR=XPOINT*XINV
C IST=I+1
C DO 18 J=IST,NPOINT
C XRATIO=X(J)*XINV
C P2=PHI2(J)
C PHI2(J)=(P1*(XRATIO-XR)+P2*(XR-1.0))/(XRATIO-1.0)
C 18 CONTINUE
C PH(L)=PHI2(NPOINT)
C IF(ABS(PH(L)).GT.ABS(PHMAX))PHMAX=PH(L)
C PHI(JMAX2,K,L)=PH(L)
C PHI(JMAX2,K,LP)=PH(L)
C 20 CONTINUE
C... AT THIS K, FIND MAXIMUM DEVIATION IN PH(L)'S; PUT RESULT IN PHAC(K).

```
ERR=0.0
DO 30 L=1,LM
    ER=1.0-PH(L)/PHMAX
    IF(ABS(ER).GT.ABS(ERR)) ERR=ER

30 CONTINUE
PHAC(K)=ERR
```

50 CONTINUE

C... THE NEXT LOOP FINDS LARGEST OF ALL DEVIATIONS IN PHI'S ON Z-PUTS VALUE IN PHICHK AND K-VALUE IN KLOCAT.

```
ERR=0.0
DO 60 K=2,KMAX1
    XX=ABS(PHAC(K))
    IF(XX.LE.ABS(ERR)) GO TO 60
    ERR=PHAC(K)
    KLOCAT=K

60 CONTINUE
PHICHK=ERR
```

```
IF(IPRINT.NE.0) WRITE(6,101)(PHAC(K),K=2,KMAX1)
IF(LMAX.EQ.1)RETURN
IF(IPRINT.NE.0) WRITE(6,102)(J,((COEF(J,M,III),M=1,8),III=1,12,JMAX)
RETURN
```

END
Vita

Name: Harold Alden Williams.


Citizenship: United States of America.

Family: Wife Barbara Sue and daughter Stephanie Rose.


Short Courses: Advanced-study course on the Foundation of Nonequilibrium Statistical Physics at Albuquerque, N.M. and NATO Advanced Study Institute on Frontiers of Nonequilibrium Statistical Physics at Santa Fe, N.M., May 20 to June 16, 1984; and University of Minnesota Supercomputer Institute Summer Workshop sponsored by NSF (included Pacific-Sierra Research Corporation course in Efficient Fortran Techniques for Vector Processors), August 1986.

Courses Taught: physics labs for pre-med, physics labs for physics majors, astronomy lab for general science, and short course in astronomy for gifted high-school students.


Research: multidimensional hydrodynamics, star formation, large scale computing, Wigner distributions, and physics in general.
PUBLICATIONS


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Harold Alden Williams

Major Field: Physics

Title of Dissertation: Star Formation, Using 3-D Explicit Eulerian Hydrodynamics

Approved:

[Signatures]

Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

[Signatures]

R. D. Hussey

N. E. Baron

Richard Haymachen

[Signatures]

S. S. Subramaniam

George E. Vaynoglou

Date of Examination:

January 13, 1988