Documentation for YREC7d

Yale Stellar Evolution Code with Rotation

PIERRE DEMARQUE, DAVID GUENTHER, AND MARC PINSONNEAULT

DECEMBER 2009

DAVID GUENTHER

Table of Contents

Introduction

The Yale Evolution Code (YREC) is a FORTRAN program that calculates the evolution of the structure of a star. The evolution is represented by a sequence of stellar models monotonically increasing in age (but not necessarily equally spaced). Each stellar model lists temperature, density, pressure, energy generation, chemical composition and, if applicable, rotation rate (along with many other variables).

NOTE: The rotation sections of the current version of the code are not up to date. Please see Pinsonneault for more information.

YREC is normally run in the background without user interaction. All the parameters that control the program are specified in NAMELIST files. The program reads a starting model (that must be supplied) and then evolves the model, writing information about the evolved models to various files as the evolution progresses. YREC is also capable of rescaling the core mass, the mass, and/or the composition of the starting model.

YREC literally embodies nearly everything we know about stellar structure and evolution (circa today), and to master the entire code requires years of study. The program, if misunderstood, is capable of producing garbage. It is, therefore, recommended that the advanced student spend some time learning how the program works. It is probably not possible to run YREC as a "black-box," unless you have an expert handy. This documentation explains how to run YREC and in some cases, how the program works.

NOTE: YREC is not in the public domain. It is a research tool originally developed at Yale University by individuals in the Department of Astronomy, using departmental resources. Permission to use or copy this program must be obtained from Pierre Demarque, David Guenther, or Marc Pinsonneault.

Some Credits (or who to bug if something goes wrong)

- Pierre Demarque - Overseer of all that is stellar evolution; Department of Astronomy, Yale University, P.O. Box 208101, New Haven, CT 06520-8101, USA; demarque@astro.yale.edu.
- David Guenther - cleaned up various sections of the code; implemented many of the modern elements; and wrote this manual; Department of Astronomy and Physics, Saint Mary's University, Halifax, NS, B3H 3C3, CANADA; guenther@ap.stmarys.ca.
- Marc Pinsonneault - rewrote much of PSEP; added rotation to produce the first YREC; Astronomy Department, Ohio State University, 174 West 18th Ave., Columbus, OH 85726-6732, USA; pinsono@astronomy.ohio-state.edu.
- Michael Prather - author of the Prather Stellar Evolution Program upon which YREC is based.
- Peter Cole - modified PSEP to follow He flash phase.
- Y.-C. Kim, Brian Chaboyer and many more deserving of mention - added new physics; improved existing physics.

New in YREC7

Yrec7d includes Ferguson opacities and OPAL95 opacities with full interpolation into tables. That is they allow interpolation of Z, as well as X, and density and temperature. Yrec7d also adds mass loss and gain routines.

YREC6 has been the reference version of YREC since 1996. YREC7, released in May 1999, contains several new options. YREC7 is backward compatible with YREC6.

- YREC7 now allows one to carry out many different runs each with their own set of CONTROL and PHYSICS NAMELIST parameters (a.k.a. *.nml1 and *.nml2 files, respectively). In YREC6 one was forced to use a single pair of files named yrec6.nml1 (CONTROL NAMELIST) and yrec6.nml2 (PHYSICS NAMELIST), located in the execution directory of YREC6. All parameters associated with the run were defined in these two files. With YREC7 it is now possible to string together runs defined by a variety of different NAMELIST pairs. YREC7 reads the NMLCONTROL NAMELIST (a.k.a. yrec7.batch). The NMLCONTROL NAMELIST contains a list of files containing the CONTROL and PHYSICS NAMELISTS which YREC7 will then read and use in sequence.

- In YREC7, a new option exists that will output a detailed listing of a model, a.k.a. the pulsation output, at user specified ages. That is, one can define a series of ages, and YREC7 will automatically output a detailed listing of the structure at exactly those ages, all the while, continuing with the standard evolutionary calculation.

- NOTE that the file naming convention for pulsation output files has been changed. Only the filename prefix is required. YREC7 will now automatically add the .pmod, .penv, and .patm suffixes. In addition, a different pulsation output format has been implemented (IPVER=4) that appends the envelope and atmosphere data to the model data file, i.e., with IPVER=4, a single file contains all the data formerly contained in the three pulsation output files. Guenther's nonradial, nonadiabatic stellar pulsation program jig8 can read this new format. Use IPVER=3 if you are running an earlier version of jig.

- In YREC7 an option to allow extrapolation in the Alexander and OPAL95 opacity tables has been implemented. In YREC6 if an opacity was sought for a temperature and density outside the edge of the opacity table the program would print out an error message and halt YREC7. In YREC7 you can specify a tolerance range, in log10, by which YREC7 will linearly extrapolate beyond the log-(normalized) temperature, log-(normalized) density edge of the table.

Principles of Operation

YREC solves the linearized equations of stellar evolution in one dimension (rotation is represented in an oblate coordinate system to maintain the 1-D nature of the code). The pressure p, temperature T, radius R, and luminosity L are solved at a given time t, as a function of mass m. A set of finite difference equations relate the pressure, temperature, radius, and luminosity, a function of mass and time, at each shell in the model.

The energy generation rate, the mean molecular weight, the density and the opacity are calculated separately. The reaction rates of all the important nuclear reactions (pp chains, CNO cycles, triple-alpha, and some light element reactions) are determined from the known nuclear cross sections. By solving the network of reactions the program determines both the nuclear photon and neutrino luminosity and the evolution of the elements involved in the reactions for each shell. The equation of state routines calculate the density and other thermodynamic variables. Once the new composition of elements is determined the mean molecular weight and the opacities are calculated. Tables of opacity, a function of temperature, density and chemical composition are interpolated to obtain the opacity in each shell of the model. The equation of

state can be calculated empirically using an approximate formula relating density to the pressure and temperature, which takes into account the degree of ionization of hydrogen and helium or it can be obtained by interpolating a set of tables.

Because pressure, density, and temperature span many orders of magnitude from core to surface in a star it is necessary to carry out the solution procedure in three separate regions: the interior, the envelope, and the atmosphere. With respect to current parameters in the stellar evolution program the interior refers to the inner 99.98% by mass of the star, the envelope refers to the outer 0.02% by mass of the star, and the atmosphere (massless) refers to the region running from an optical depth of $\tau=2/3$ at the base of the photosphere (note that the optional Krishna Swamy model atmosphere routine places the base of the atmosphere at $\tau=0.31215633$), to $\tau=10^{-5}$ near the temperature minimum above which the temperature rises again. The stellar evolution program solves the equations of stellar structure in the interior and matches the solution to the solution in the envelope (see details in following paragraphs). The atmosphere is used to provide the outer boundary condition and is necessary to fix the radius of the star.

Again, the nuclear reaction rates, the chemical evolution, and the equation of state are not directly connected to the stellar evolution equations. They are calculated separately and their effects on the model are introduced into the structure calculation via changes to the mean molecular weight, changes to the relation between temperature, pressure, and density, and changes to the entropy term ($T\mathrm{d}S/\mathrm{d}t$, which appears in the conservation of energy equation). Experience has shown that the stability of the evolutionary calculation depends on the order and manner in which these contributions are made.

The basic solution procedure is a follows: given a time-step Δt and an initial guess model at time t, corrections to the pressure, temperature, and luminosity are calculated using the finite difference equations of stellar evolution for the time $t+\Delta t$. These corrections are added to the physical variables of the interior model. This corrected interior model is then used as the initial guess, replacing the previous initial guess model, in an iterative procedure which ultimately solves the stellar evolution equations. After each iteration the corrections are tested to see if they are below the user specified tolerances. When they are the interior is considered solved at the new time $t+\Delta t$. The evolution is said to "blow up" or "fail to converge" if after a user specified number of iterations the model's corrections still exceed the requested tolerances. The evolutionary run must be restarted with smaller time-steps or with one of a million other possible adjustments. Note there is an option in the code to calculate the chemical composition and nuclear reaction rates at the end of the time-step ($t+\Delta t$) so that they are self-consistent; otherwise the rates at the beginning of the time-step are used to calculate the composition at the end of the time-step.

The interior is solved assuming that the outer boundary conditions correspond to the vanishing of the pressure and temperature. It is, therefore, necessary to solve the outer envelope in detail in order to properly determine the star's radius and luminosity. This is done by integrating inward from the top of the star's atmosphere. Three guess integrations are performed. The three guesses are designed to surround the surface values of L and T of the final solution. A gray atmosphere is integrated using the Eddington T-τ relation or the Krishna Swamy T-τ relation (a fit to the Sun). The atmosphere integration extends from about $\tau=10^{-5}$ to $\tau=2/3$ (the photosphere is defined at a different τ for the K-S T-τ relation). At $\tau=2/3$ the pressure and temperature are determined. Note that because the atmosphere does not contribute to the luminosity of the star the interior integration provides the luminosity of the star at the surface, hence, P, T, L, M and R are known at this stage. The $\tau=2/3$ values of P, T, L, M, and R are then used to integrate the stellar structure equations inwards. Again, three such integrations are performed which hopefully will encompass the true solution. The three solutions are interpolated so that at the 99.998% mass radius point, the interior solution smoothly matches the inward interpolated envelope solution. If the three inward solutions lie to one side of the matching point of the interior then a new trio of

envelope and atmospheres are calculated (the process of guessing a new set of envelope-atmosphere integrations is called triangle flipping because, typically, only one or two new T, L pairs need to be calculated). The interpolated solution is then used to give the surface temperature and pressure of the star.

NOTE: YREC can determine the conditions at the base of the atmosphere by looking into tables obtained from Kurucz's model atmospheres, thereby, avoiding the approximations of the gray Eddington atmosphere calculation.

The program then calculates a new time-step and the procedure is repeated.

The user is given control over all aspects of the solution procedure and can control everything from the number of iterations to the size of the triangle in the T - L plane defined by the envelope-atmosphere trio of guess solutions. Experience shows that these values must be tweaked for different stellar evolutionary calculations to keep the path to solution from blowing up. This documentation will give examples of best values and will describe what happens when you change these values.

The Program Flow

For advanced users only. Please refer to Appendix A of Prather's thesis and to the code itself for more details. In particular you should trace through the main program as you read through the following description.

1. Specify physical constants. Read in opacity tables, Fermi integral tables, and other tables. (subroutine SETUPS and PARMIN called from MAIN)

Begin NAMELIST Loop

2. Read in the first pair of NAMELIST files, CONTROL and PHYSICS (*.nml1 and *.nml2), as specified in "yrec7.batch."

Begin Kind Card Loop: a sequence of different runs

3. Read in the initial or starting model. This can be a ZAMS model, a pre-main-sequence model or an evolved or rescaled model previously calculated and stored by YREC. (subroutine STARIN called from MAIN)

4. If using time-step cutting (LKUT=.TRUE.) then store initial model (in FLAST model output file) for use by time-step cutting routines. (subroutine WRTLST called form MAIN)

5. Locate edges of convection zones and, if applicable, the extent of the hydrogen burning shell. (subroutine FINDSH called from MAIN)

6. Determine the time-step Δt for the $t+\Delta t$ model (i.e. next model). The time-step is determined from a number of criteria (< 1.5 times previous time-step, structure changes, angular velocity changes, amount of hydrogen burning, amount of helium burning) which can be specified by the user. The time-step can also be set to a fixed value. (subroutine HTIMER called from MAIN).

7. Zero the entropy terms. (in MAIN)

Begin Run Loop: evolve or rescale N models

8. If time-step cutting is on and previous model diverged (LARGE=.TRUE) then cut time-step. (subroutine STARIN called from MAIN)

9. Determine location of convection zone edges and edges of mixed regions with overshoot. (subroutine NCONVEC called from subroutine NEWMIX called from MAIN)

10. Determine reaction rates in each shell. (subroutine NRATE (similar to ENGEB except no derivatives calculated) called from subroutine NEWMIX called from MAIN)

11. Calculate the new abundances in each radiative shell and then each convective region. Also homogenize the composition in the convective regions, i.e. mix them. (subroutine NKEMCOM called from subroutine NEWMIX called from MAIN)

12. Get rates for the nuclear reactions based on the time = t model to determine H, Z, C^{12}, and O^{16} abundances. Not used to get nuclear luminosities of model, just abundances. (subroutine ENGEN called from NEWMIX called from MAIN)

13. If semi-convection turned on (LSEMIC=.TRUE.) then determine extent of semi-convection. (subroutine SCONVEC called from subroutine NEWMIX called from MAIN)

14. Mix the convection zones, again. (subroutine NEWMIX called from MAIN)

15. If diffusion turned on (LDIFY=.TRUE.) then calculate gravitational settling diffusion (subroutine GRSETT called from subroutine NEWMIX called from MAIN)

16. Rezone shells, unless He flash calculations (LKUTHE=.TRUE.). (subroutine HPOINT called from MAIN)

17. If following light element abundances (LEXCOM=.TRUE.) then need to store depth of convection zone (after rezoning) for time t model. (subroutine NCONVEC called from MAIN)

18. If LNEWS=.FALSE. then zero entropy terms, otherwise use rate of change of previous model to guess new entropy terms and structure (gravity term is zeroed, regardless). (in MAIN)

First level of iteration

19. Do not calculate new envelope triangle unless this is the first model of the run. If a new envelope triangle is to be calculated, begin with a call to subroutine SURFBC which sets up a triangle (in the H-R diagram) of three surface temperature, luminosity pairs that are likely to surround the final converged model. Subroutine ENVINT is then called three times, once for each temperature, luminosity pair. ENVINT starts at the top of the atmosphere and integrates the user specified T-τ relation (default is Eddington) toward $\tau=2/3$, i.e. the surface. ENVINT then integrates the structure equations shooting from the atmosphere solution at $\tau=2/3$ to the outermost mass shell of the interior. (subroutines SURFBC and ENVINT called from subroutine CRRECT called from MAIN).

20. Calculate the coefficients of the stellar structure equations at each shell. (subroutine COEFFT called from subroutine CRRECT called from MAIN)

    For each shell:

    - determine ρ and other thermodynamic variables and their derivatives from P and T (equation of state routines called from subroutine COEFFT called from subroutine CRRECT called from MAIN);

- determine opacity (subroutine GETOPAC called from subroutine COEFFT called from subroutine CRRECT called from MAIN);
- determine temperature gradients (subroutine TPGRAD or TPDPRE called from subroutine COEFFT called from subroutine CRRECT called from MAIN);
- determine nuclear reaction luminosities (if (LNEWE=.TRUE. subroutine ENGEB called from subroutine COEFFT called from subroutine CRRECT called from MAIN (new nuclear reaction rates), otherwise subroutine ENGEN called from subroutine COEFFT called from subroutine CRRECT called from MAIN(old rates)).

21. With coefficient matrix and finite difference form of structure equations solve system of linear equations (four equations times N shells for a total of 4N equations) for corrections, i.e. values to be added to previous model's variables to get model at time $t+\Delta t$. The linear equations are solved by iteration until the corrections converge to stable values which are less than user specified tolerances. (subroutine CRRECT called from MAIN)

Second level of iteration

22. Calculate surface boundary conditions. This begins with a call to subroutine SURFBC which sets up a triangle (in the H-R diagram) of three surface temperature, luminosity pairs that are likely to surround the final converged model. Subroutine ENVINT is then called three times, once for each temperature, luminosity pair. ENVINT starts at the top of the atmosphere and integrates the user specified T-τ relation (default is Eddington) toward $\tau=2/3$, i.e. the surface. ENVINT then integrates the structure equations shooting from the atmosphere solution at $\tau=2/3$ to the outermost mass shell of the interior. (subroutines SURFBC and ENVINT called from subroutine CRRECT called from MAIN).

23. Calculate the coefficients of the stellar structure equations at each shell (subroutine COEFFT called from subroutine CRRECT called from MAIN). Similar to step (21).

Optional Third level of iteration.

Only performed if greater consistency between abundances and structure wanted (LNEWM=.TRUE.)

24. Do not calculate surface boundary conditions - as this can introduce instabilities.

25. Calculate evolved abundances of model, i.e. abundances based on the converged $t+\Delta t$ model generated after the second level of iteration. That is to say, the nuclear rates of the $t+\Delta t$ are used to determine the new composition. With the new composition, new rates are calculated and the new rates are used to determine an improved estimate of the new composition. The process is iterated until self-consistency is obtained between the composition and nuclear reaction rates. (subroutine NEWMIX called from subroutine CRRECT called from MAIN)

26. Calculate the coefficients of the stellar structure equations at each shell (subroutine COEFFT called from subroutine CRRECT called from MAIN). Similar to step (21).

Optional Fourth level of iterations.

One further iteration to improve consistency between abundances and structure. Only required for high precision models. Do not use with semi-convection (i.e. H.B. stars) as oscillations will develop.

27. Do not calculate surface boundary conditions - as can introduce instabilities.

28. Calculate evolved abundances of model, i.e. abundances based on the converged $t+\Delta t$ model generated after the second level of iteration. That is to say, the nuclear rates of the $t+\Delta t$ are used to determine the new composition. With the new composition, new rates are calculated and the new rates are used to determine an improved estimate of the new composition. The process is iterated until self-consistency is obtained between the composition and nuclear reaction rates. (subroutine NEWMIX called from subroutine CRRECT called from MAIN)

29. Calculate the coefficients of the stellar structure equations at each shell (subroutine COEFFT called from subroutine CRRECT called from MAIN). Similar to step (21).

30. If model did not converge either stop or if time-step cutting is enabled go to step (8). (in MAIN)

Converged model

31. Once again, re-mix convection zones. (subroutine MIXCX called from MAIN)

32. If calculating rotating models determine new rotation curve here. (subroutine FINDW called from MAIN)

33. Locate edges of hydrogen burning shell and convection zones. (subroutine FINDSH called from MAIN)

34. If following light elements (LEXCOM=.TRUE.) then calculate evolved (i.e. burnt) abundances of light elements. (subroutine NCONVEC called from MAIN) For lithium calculate its evolved burning rate. (subroutine LIRATE (old) called from MAIN or LIRATE88 (new) called from MAIN) And then calculate its abundance. (subroutine LIBURN called from MAIN)

35. Determine the time-step (see step (6)). (subroutine HTIMER called from MAIN)

36. Write out requested information about current model. (subroutine WRTOUT called from MAIN). Note that there are other output routines that serve specific purposes, such as the pulsation output. The pulsation output combines interior, envelope, and atmosphere data, hence is useful in other contexts as well.

Continue Run Loop: evolve until last model

Continue Kind Card Loop: begin next run as specified by kind cards

Continue NAMELIST Loop: do next pair of NAMELISTs

An example of the NAMELIST files

The NAMELIST files contain all of the user adjustable variables which control the operation of YREC. This section describes the function of these parameters. The NAMELIST is a feature of VAX FORTRAN and is not part of the FORTRAN 77 ANSI standard upon which much of the code in YREC is based. Most modern FORTRAN compilers do, though, include such a feature.

YREC defines three different NAMELISTs: NMLCONTROL, CONTROL, and PHYSICS. The NMLCONTROL NAMELIST is hard coded, in main.f, to be the file "yrec7.batch." It contains a list of the file names containing the other two NAMELISTs. The CONTROL NAMELIST is contained in a file, usually designated by the suffix: .nml1, and the PHYSICS NAMELIST is contained in a file, usually designated by the suffix: .nml2.

The following is an example of a yrec7.batch file containing the NMLCONTROL NAMELIST:

```
 $NMLCONTROL

 NUMNML=3

 FNML1(1)='E:\dguenther\research\MOST\nml\sbd06.nml1'
 FNML2(1)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

 FNML1(2)='E:\dguenther\research\MOST\nml\sbd07.nml1'
 FNML2(2)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

 FNML1(3)='E:\dguenther\research\MOST\nml\sbd08.nml1'
 FNML2(3)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

 $END
```

The $CONTROL NAMELIST contains commonly adjusted parameters such as those that determine what data is to be input and output. An example of the $CONTROL NAMELIST file follows:

```
$CONTROL

 DESCRIP(1)='M=m100 Z=0.0200 Sun'
 DESCRIP(2)='Grey atmos, Ferg & new OPAL95 opac, OPAL2001 eos Y&Z diff'

 LCALS = .FALSE.
 TOLL = 2.0D-5
 TOLR = 3.0D-5
 LINCLUDEZX = .FALSE.
 TOLZX = 1.0D-4
 ZOVERX = 0.0244D0

 LCALST=.FALSE.
 XLS=0.4937D0
 XLSTOL=0.0010D0
 LTEFF=.TRUE.
 STEFF=4300.0D0
 LADJX=.FALSE.

 LZRAMP = .FALSE.

 NUMRUN=2

 KINDRN(1)=2
 LFIRST(1)=.TRUE.
 RSCLM(1)= -1.670D0
 RSCLCM(1)=-0.10D0
 RSCLX(1)=0.700D0
 RSCLZ(1)=0.0200D0
 RSCLZC(1)=-0.0001D0
 RSCLZM1(1)=-0.20D0
 RSCLZM2(1)=-0.21D0
 NMODLS(1)= 1
 XENV0A(1)=0.700D0
 ZENV0A(1)=0.020D0
 CMIXLA(1)= 1.8D0
 LSENV0A(1) = .FALSE.
 SENV0A(1) = 1.0D-8

 KINDRN(2)=1
 LFIRST(2)=.FALSE.
 ENDAGE(2)=-4.55D9
 SETDT(2)= -0.091D9
```

```
NMODLS(2)= 500000
XENV0A(2)=0.70D0
ZENV0A(2)=0.020D0
CMIXLA(2)= 1.8D0
LSENV0A(2) = .FALSE.
SENV0A(2) = 1.0D-8

LZAMSSTOP(2)=.FALSE.
MBEYONDZAMS(2)=20
GRAVTH(2)=0.01D0

LTRACK=.TRUE.
ITRVER=3

LRWSH=.FALSE.
LSCRIB=.FALSE.
NPRT1=1
NPRT2=1
NPRTPT=1
LPSHLL=.FALSE.
LCONZO=.FALSE.
LCHEMO=.FALSE.
LJOUT=.FALSE.
LPRTIN=.FALSE.
NPENV=-1

LCORR=.FALSE.
NPOINT=-1
LSTORE=.FALSE.
NPUNCH=-1
LSTPCH=.FALSE.

LPULSE=.FALSE.
IPVER=4
LPOUT=.FALSE.
POMAX=8.0E-3
POA=1.0D0
POB=1.0D1
POC=0.0D0

LAGES=.FALSE.
NAGES = 1
AGES(1)= 0.05

LBNIN=.FALSE.
LBNOUT=.FALSE.

NBN=1000
NLST=0

LDEBUG=.FALSE.

LMILNE=.FALSE.

LISO = .FALSE.
LOPT1 = .FALSE.
LOPT2 = .FALSE.
LOPT3 = .FALSE.

FFIRST='/Volumes/HD3/RESEARCH/evolve/zams/forpulsation/z20y27m100.zams'

FLAST='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb.last'
FMODPT='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb.full'
FSTOR='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb.store'
FTRACK='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb.track'
```

```
FSHORT='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb.short'

FPMOD='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb'
FPENV=' '
FPATM=' '

FOPALE='/Volumes/HD3/RESEARCH/evolve/eos_OPAL2001/OPAL2001eos.0200'

LFERG05 = .TRUE.
FFERG05 = '/Volumes/HD3/RESEARCH/evolve/opacities/FERGUSON/GN93ferg'

LALEX05=.FALSE.
ZALEX051=0.0200D0
ZALEX052=0.0300D0
OPECALEX05(1) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X000'
OPECALEX05(2) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X010'
OPECALEX05(3) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X020'
OPECALEX05(4) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X035'
OPECALEX05(5) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X050'
OPECALEX05(6) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X070'
OPECALEX05(7) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X080'
OPECALEX05(8) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X090'
OPECALEX05(9) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX05/GN93Alex05X095'

LALEX95=.FALSE.
TOLALEX=0.00D0
ZALEX1=0.0200D0
ZALEX2=0.0300D0
OPECALEX(1) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X00'
OPECALEX(2) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X01'
OPECALEX(3) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X02'
OPECALEX(4) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X035'
OPECALEX(5) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X05'
OPECALEX(6) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X07'
OPECALEX(7) = '/Volumes/HD3/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X08'

LOPAL95=.TRUE.
LNEWOPAL95=.TRUE.
TOLOPAL95=0.00D0
ZOPAL951=0.0200D0
ZOPAL952=0.0300D0
FLIV95 = '/Volumes/HD3/RESEARCH/evolve/opacities/OPAL95/GN93hz'

LKUR90=.FALSE.
ZKUR1=0.0300
ZKUR2=0.040
FKUR='/Volumes/HD3/RESEARCH/evolve/opacities/KURUCZ/kapp00.ross'
FKUR2='/Volumes/HD3/RESEARCH/evolve/opacities/KURUCZ/kapp03.ross'

LOPAL92=.FALSE.
FLLDAT='/Volumes/HD3/RESEARCH/evolve/opacities/OPAL92/Z017G93.OPAL'
ZOPAL1=0.03000D0
ZOPAL2=0.023D0
FOPAL2='/Volumes/HD3/RESEARCH/evolve/opacities/OPAL92/Z023G93.OPAL'

LLAOL89=.FALSE.
ZLAOL1=0.0300D0
ZLAOL2=0.02D0
FLAOL='/Volumes/HD3/RESEARCH/evolve/opacities/LAOL/Z0190AG91.DBGLAOL'
FLAOL2='/Volumes/HD3/RESEARCH/evolve/opacities/LAOL/Z02AG91.DBGLAOL'

LPUREZ=.TRUE.
FPUREZ='/Volumes/HD3/RESEARCH/evolve/opacities/LAOL/PURECO.DBGLAOL'

FLSTBN='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb'
```

```
FSTOBN='/Volumes/HD3/RESEARCH/gianttests/output/d3_gb'
FFSTBN=''
FISO=''
FVG=''
FDELOP=''

FDYN=''
FSNU=''
FSCOMP=''
FDEBUG=''
FMILNE=''

FFERMI='/Volumes/HD3/RESEARCH/evolve/misc/fermi.tab'

FMHD1=''
FMHD2=''
FMHD3=''
FMHD4=''
FMHD5=''
FMHD6=''
FMHD7=''
FMHD8=''

LMLOSS=.FALSE.
CMLL=0.0D0
CMLR=0.0D0
CMLM=0.0D0
CMLG=0.0D0
CMLETA=0.0D0
CMLC=-11.5D0
ETAML=1.0D0
FRACTML=0.3D0
TINSTLO=0.0D0
TINSTHI=0.0D0

$END


! Default values in ().
! Description fields at head of model output
DESCRIP(1)='1 M_sun'
DESCRIP(2)='Alex & OPAL95 opacities OPAL95 EOS Y and Z diffusion'

LCALS=.FALSE.         ! (F) T-then iterate runs to converge to the
                      ! solar luminosity and radius and, optionally, Z/X
                      ! at end of run model.
TOLR=1.0D-6           ! Log R/Rsun must be within TOLR of 0.0
TOLL=1.0D-7           ! Log L/Lsun must be within TOLL of 0.0
LINCLUDEZX = .FALSE.  ! If T then also try to tune Z/X to ZOVERZ within
                      ! tolerance TOLZX
TOLZX = 1.0D-4        ! Z/X must be within TOLZX of ZOVERX
ZOVERX = 0.0244D0     ! Z/X

LCALST=.FALSE.        ! (F) T-then iterate runs to converge a stellar
                      ! model at specified luminosity and radius (Teff).
                      ! Adjusts Y (and age) to get correct model.
XLS=0.4937D0          ! luminosity wanted for stellar model (L/Lsun).
XLSTOL=0.001D0        ! accuracy of model fit to specified luminosity
LTEFF=.TRUE .         ! (F) T-then tune model to specified Teff
                      ! F-then tune model to specified radius
STEFF=5300.0D0        ! desired Teff of model (used if LTEFF=T) (K)
SR=1.0D0              ! desired radius of model (R/Rsun) (used if LTEFF=F)
LADJX=.FALSE.         ! T-then adjust X to produce tuned model
                      ! F-then adjust alpha to produce tuned model
```

```
LZRAMP=.FALSE.        ! Use RSCLZC (center to RSCLZM1) then
                      ! ramp Z from RSCLZC at RSCLZM1 to
                      ! RSCLZ at RSCLZM2. Must input opacities
                      ! tables for interior Z

NUMRUN=2              ! number of runs; maximum is 50

KINDRN(1)=2           ! type of run: 1 = evolve, 2 = rescale,
                      ! 3 = evolve and rescale (for pre-ms rescaling)
LFIRST(1)=.TRUE.      ! T=use "first.mod", F=use last mod. of prev. run

RSCLM(1)= -0.95D0     ! if .GT. 0 rescale mass of LFIRST model to RSCLM (Msun)
RSCLCM(1)=-0.10D0     ! if .GT. 0 rescale X of LFIRST model to RSCLX
RSCLX(1)=-0.680D0     ! if .GT. 0 rescale Z of LFIRST model to RSCLZ
RSCLZ(1)=-0.020D0     ! if .GT. 0 rescale core mass of LFIRST model to RSCLCM
RSCLZC(1)=-0.0001D0   ! if LZRAMP=T rescale Z in core
RSCLZM1(1)=-0.20D0    ! if LZRAMP=T rescale Z in core
RSCLZM2(1)=-0.21D0    ! if LZRAMP=T rescale Z in core

NMODLS(1)= 1          ! number of models to "relax" rescaled model.
XENV0A(1)=0.680D0     ! ZAMS X
ZENV0A(1)=0.020D0     ! ZAMS Z
CMIXLA(1)= 2.0D0      ! mixing length parameter
LSENV0A(1) = .FALSE.  ! (F) if T then reset outer mass of Henyey integration
SENV0A(1) = 1.0D-8    ! value to set outer mass of Henyey integration

KINDRN(2)=1           ! type of run: 1 = evolve, 2 = rescale
LFIRST(2)=.FALSE.     ! T=use "first.mod", F=use last mod. of prev. run
ENDAGE(2)=-4.5E9      ! if nonzero then stop run when age (yr) equals ENDAGE
SETDT(2)= -0.09E9     ! if nonzero then force time-step to SETDT (yr)
NMODLS(2)= 500        ! number of models in run
XENV0A(2)=0.680D0     ! ZAMS X
ZENV0A(2)=0.020D0     ! ZAMS Z
CMIXLA(2)= 2.0D0      ! mixing length parameter
LSENV0A(1) = .FALSE.  ! (F) if T then reset outer mass of Henyey integration
SENV0A(2) = 1.0D-8    ! value to set outer mass of Henyey integration

LZAMSSTOP(2)=.FALSE.  ! T then halt pre-main sequence evolution after reaching ZAMS
MBEYONDZAMS(2)=20     ! evolve MBEYONDZAMS models after reaching ZAMS
GRAVTH(2)=0.01D0      ! ZAMS defined where gravitational energy is GRATH fraction of
                      ! total energy, i.e., nuclear energy dominates.

LTRACK=.FALSE         ! (T) TRACK output: Teff, L, etc. for each model
ITRVER=3              ! the version of the *.track file 1 = default
                      ! or standard format
                      ! 2 = include rotation variables
                      ! 3 = one line basic info

LRWSH = .FALSE.       ! If T then rewinds ISHORT after each model to keep
                      ! file small.  Only last diagnostics should exist.

LSCRIB=.TRUE.         ! (T) output listing to IMODPT
NPRT1=1               ! output basic model info to IMODPT every NPRT1 models
NPRT2=1               ! output extra model info to IMODPT every NPRT2 models
NPRTPT=1              ! (3)print every NPRTPT points of model to IMODPT
LPSHLL=.FALSE.        ! (F) output H-burning shell info to IMODPT
LCONZO=.FALSE.        ! (F) output details of conv. zone to IMODPT
LCHEMO=.FALSE.        ! (T) output composition info to IMODPT
LJOUT=.FALSE.         ! (F) output rotation info to IMODPT
LPRTIN=.FALSE.        ! (F) output instability info
NPENV=1               ! (3) output env. info every NPENV times env. calculated
LCORR=.FALSE.         ! (T) print largest P,T,R corrections to shell
NPOINT=-1             ! (1) output subrout. HPOINT grid every NPOINT models

LSTORE=.FALSE.        ! (T) model output to ISTOR and last model to ILAST
```

```
NPUNCH=-1             ! output every NPUNCHth model to ISTOR
LSTPCH=.FALSE.        ! (T) store last model in ISTOR

LPULSE=.FALSE.        ! (F) output model, envelope, atmosphere info.
                      ! for pulsation
IPVER=4               ! version of pulsation output; unless you have a
                      ! reason to do so use latest version (4).
                      ! Version 1 and 2 are obsolete versions that did not
                      ! contain as many significant digits in output. Version
                      ! 3 has maximum number of significant digits. Version
                      ! 4 (must have jig8 to read this format) is identical to
                      ! version 3 except that *.patm and *.penv output are
                      ! appended to *.pmod file (rather be output to their
                      ! own distinct files.
LPOUT=.FALSE.         ! (F) if T then will dump numbered pulsation output
                      !  and *.short file info when
                      !  track has moved arclength POMAX in HR-diagram
                      ! The arclength is defined by:
                      ! LEN=(POA*Delta Log(L/Lsun))**2
                      !    +(POB*Delta Log(Teff))**2
                      !    +(POC*Delta Age)**2
                      ! where Delta is the model to model difference
                      ! in that variable.
POMAX=0.15D-3         ! (0.01D0) see LPOUT.
POA=1.0D0             ! (1.0D0) see LPOUT.
POB=10.0D0            ! (10.0D0) see LPOUT.
POC=0.0D0             ! (0.0D0) see LPOUT. Have implemented a superior age criteria below.

LAGES=.FALSE.         ! If T then output data for pulsation and *.short info
                      ! at the exact ages specified in the AGES(NAGES) vector.
NAGES = 14            ! Number of ages specified in the AGES vector. Maximum number
                      ! is coded to be 500.
AGES(1)=1.0D0         ! Array of ages in Gyr, where pulsation and *.short data will
                      ! be output
AGES(2)=2.0D0
AGES(3)=3.0D0
AGES(4)=4.0D0
AGES(5)=5.0D0
AGES(6)=6.0D0
AGES(7)=7.0D0
AGES(8)=7.01D0
AGES(9)=7.02D0
AGES(10)=7.021D0
AGES(11)=7.022D0
AGES(12)=7.023D0
AGES(14)=7.024D0

LBNIN=.FALSE.         ! (F) read binary formatted initial model
LBNOUT=.TRUE.         ! (T) store last model and intermediate models
                      ! in binary format
NBN=1000              ! store every NBN models in binary format
NLST=2                ! save last NLST models in binary format

LDEBUG=.FALSE.        ! (F) debugging output to IDEBUG

LMILNE=.FALSE.        ! (F) output Milne invariants U, V, etc to IMILNE

LISO = .FALSE.        ! If T output data for isochrone construction to IISO
LOPT1 = .FALSE.       ! If T output optional info #1 to IISO (not used)
LOPT2 = .FALSE.       ! If T output optional info #2 to IISO (not used)
LOPT3 = .FALSE.       ! If T output optional info #3 to IISO (not used)


! Starting (input) model.
FFIRST='/RESEARCH/evolve/zams/forpulsation/z20y27m100.zams'
```

```
! last model computed for each run. File format is same as starting
! model
FLAST='/RESEARCH/RUN/TEST/m1000_a05_e95_diff.last'
!
! Detailed model structure output which can be controlled through variables
! in CONTROL NAMELIST. The output is nicely formatted.
FMODPT='/RESEARCH/RUN/TEST/m1000_a05_e95_diff.full'
!
! obsolete. Model file in format same as input model.
FSTOR='/RESEARCH/RUN/TEST/m1000_a05_e95_diff.store'
!
! General information about model as it evolves. HR diagram info.
FTRACK='/RESEARCH/RUN/TEST/m1000_a05_e95_diff.track'
!
! Detailed general model information as it evolves. One numbered short
! file is produced with each *.PMOD file when LPULSE=.TRUE.
FSHORT='/RESEARCH/RUN/TEST/m1000_a05_e95_diff.short'

! numbered model file extending from center of star to temperature
! minimum at surface. FPENV and FPATM obsolete. ".pmod" is automatically
! appended to FPMOD.
! Pulsation output in a format
! that can be read by pulsation program
! *.pmod contains pmod, patm, and penv info in
! version 4, that is, FPENV and FPATM are not
! used when output version 4 pulse info.
! When LAGE=.TRUE. or LPOUT=.TRUE. then sequentially
! numbered files will be created using FPMOD as the
! prefix and .pmod, and .short as the suffix. That is
! FPMOD, FPENV, and FPATM should not contain the suffixes
! as YREC will automatically add them.
FPMOD='/RESEARCH/RUN/TEST/m1000_a05_e95_diff'
FPENV=' '
FPATM=' '

! OPAL equation of state tables. Warning there are two incompatible
! formats, the 1995 version and the more modern 2001 version. The
! flag OPAL2001EOS flag in PHYSICS NAMELIST determines which equation of state
! is used. Must set LOPALE flag in PHYSICS NAMELIST to use OPAL equation of state
! tables.
FOPALE='/RESEARCH/evolve/eos_OPAL1995/OPALeos.0200'

! Use flags to turn on or off one surface, or low T, opacity and one interior
! opacity

LFERG05 = .TRUE. ! (F) T use Ferguson 2005 low temp opacities
! A single table is read for all compositions and interpolation is done
! to Z (and X).
FFERG05 = '/Volumes/HD3/RESEARCH/evolve/opacities/FERGUSON/GN93ferg'

! Alexander 2005 low T opacity tables.
! Tables will be interpolated to ZALEX051 and ZALEX052
! when required, e.g., Z diffusion.
LALEX05=.TRUE.     ! T use Alexander 2005 tables
ZALEX051=0.0200D0  ! Z value. Should correspond to model Z
ZALEX052=0.0300D0  ! If Z varying, then specify second Z so code will
                   ! interpolate between to Z tables.
OPECALEX05(1) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X000'
OPECALEX05(2) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X010'
OPECALEX05(3) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X020'
OPECALEX05(4) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X035'
OPECALEX05(5) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X050'
OPECALEX05(6) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X070'
OPECALEX05(7) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X080'
```

```
OPECALEX05(8) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X090'
OPECALEX05(9) = '/RESEARCH/evolve/opacities/ALEX05/Alex05X095'

! Alexander 1995 low T opacity tables.
LALEX95=.FALSE.
TOLALEX=0.00D0
ZALEX1=0.0200D0
ZALEX2=0.0300D0
OPECALEX(1) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X00'
OPECALEX(2) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X01'
OPECALEX(3) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X02'
OPECALEX(4) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X035'
OPECALEX(5) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X05'
OPECALEX(6) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X07'
OPECALEX(7) = '/RESEARCH/evolve/opacities/ALEX/OPACALEXANDER.X08'

! Kurucz 1990 Low temperature opacity tables
! specific table must be read which corresponds to
! specified Z
LKUR90=.FALSE.        ! T - use Kurucz opacity tables in
                      ! low T regime
ZKUR1=0.019           ! Z of table
ZKUR2=0.038           ! if required, second Z table
FKUR='/RESEARCH/evolve/opacities/KURUCZ/kapp00.ross'
FKUR2='/RESEARCH/evolve/opacities/KURUCZ/kapp03.ross'

! OPAL 1995 interior opacities (dense grid)
! Set LNEWOPAL95 so entire table read in and interpolation Z
! is done, otherwise:
! tables will be interpolated to ZOPAL951 (and ZOPAL952
! when required, e.g., Z diffusion).
LOPAL95=.TRUE.        ! T - use OPAL 95 opacities
LNEWOPAL95=.TRUE.     ! (F) use full interpolation routines including Z
TOLOPAL95=0.0D0       ! allow opacity extrapolation from edge
                      ! of tables by this amount.  Amount is
                      ! in the log.
ZOPAL951=0.0200D0     ! interpolate tables to this Z
ZOPAL952=0.0300D0     ! if required, second Z table
FLIV95 = '/RESEARCH/evolve/opacities/OPAL95/GN93hz'

! OPAL 1992 interior opacities
! specific table must be read in corresponding to
! specified Z
LOPAL92=.FALSE.       ! T - use OPAL 92 opacities
FLLDAT='/RESEARCH/evolve/opacities/OPAL92/Z0200.OPAL'
ZOPAL1=0.02000D0
ZOPAL2=0.03D0
FOPAL2='/RESEARCH/evolve/opacities/OPAL92/Z0300.OPAL'

! LAOL89 interior opacities (DBGALOL format)
! specific table must be read in corresponding to
! specified Z
LLAOL89=.FALSE.       ! T - use LAOL opacities
ZLAOL1=0.0300D0
ZLAOL2=0.02D0
FLAOL='/RESEARCH/evolve/opacities/LAOL/Z0190AG91.DBGLAOL'
FLAOL2='/RESEARCH/evolve/opacities/LAOL/Z02AG91.DBGLAOL'

! LAOL89 opacity for pure CO composition. This table is
!    required for HB evolution when Z changes. Use this
!    regardless of the other source of opacity tables
LPUREZ=.TRUE.     ! T - read in this table
FPUREZ='/RESEARCH/evolve/opacities/LAOL/PURECO.DBGLAOL'

! model output in binary form.
```

```
 FLSTBN=''
 FSTOBN=''
 FFSTBN=''

 ! output for constructing isochrones
 FISO=''

 ! output for constructing isochrones
 FVG=''

 ! file containing list of scaling factors for
 ! for opacities.
 FDELOP=''

 ! rotation output
 FDYN=''

 ! solar neutrino output
 FSNU=''

 ! extended composition (light elements) output
 FSCOMP=''

 ! debugging output
 FDEBUG=''

 ! milne invariants output (Test your knowledge, find out what these are
 ! what they are used for.)
 FMILNE=''

 ! fermi integral tables input
 FFERMI='/RESEARCH/evolve/misc/fermi.tab'

 ! Mihalas Hummer Däppen EOS tables (superseded by OPAL EOS)
 FMHD1=''
 FMHD2=''
 FMHD3=''
 FMHD4=''
 FMHD5=''
 FMHD6=''
 FMHD7=''
 FMHD8=''

 LMLOSS=.FALSE. ! (F) T compute mass loss/gain according at a rate
 ! set by follwing parameters

 CMLL=0.0D0
 CMLR=0.0D0
 CMLM=0.0D0
 CMLG=0.0D0
 CMLETA=0.0D0
 CMLC=-11.5D0
 ETAML=1.0D0
 FRACTML=0.3D0
 TINSTLO=0.0D0
 TINSTHI=0.0D0
```

The $PHYSICS NAMELIST contains variables that tune or tweak the operational parameters of the code. Once set, these parameters rarely require adjustment. An example $PHYSICS NAMELIST follows:

```
$PHYSICS

 LDIFY = .TRUE.
 LDIFZ = .TRUE.
```

```
LTHOUL = .TRUE.
LTHOULFIT = .FALSE.
ILAMBDA = 4
FGRY = 1.0D0
FGRZ = 1.0D0
DT_GS = 0.1D0
XMIN = 1.0D-3
YMIN = 1.0D-3
GRTOL = 5.0D-6
NITER_GS = 10

KTTAU=0

LDH = .FALSE.
ETADH0 = -1.0D0
ETADH1 = 1.0D0

LOPALE = .TRUE.
LOPAL2001EOS = .TRUE.

LMHD = .FALSE.
LOPALD = .FALSE.

TOLLAOL = 10.0
LDELOP = .FALSE.

LNEWE = .TRUE.
LSNU=.TRUE.
LNEWM = .TRUE.
LNEWS = .TRUE.

NITER1=2
NITER2=40
NITER3=0
NITER4=0

LDPREE=.FALSE.
MIXKIND=0
MIXVAR=0
LHPZ=.TRUE.
LQEQ1=.FALSE.

HPBETA(1) = 9.229979D0
HPBETA(2) = -2.202276D-6
HPBETA(3) = 1.759652D-13
HPBETA(4) = -5.246017D-21
HPBETA(5) = 5.350172E-29
HPBETA(6) = 0.0D0
HPBETA(7) = 0.0D0
HPBETA(8) = 1.3D5
HPBETA(9) = 5.0D-2
HPBETA(10) = 10.0D0

LSEMIC = .FALSE.
DPENV=1.0
LOVSTC=.FALSE.
ALPHAC=0.0
LOVSTE=.FALSE.
ALPHAE=0.5
LOVSTM=.FALSE.
ALPHAM=0.0

LADOV = .FALSE.

LNSTDMX=.FALSE.
```

```
PNSTDMX(1) = 1.0D0
PNSTDMX(2) = 0.50D0
PNSTDMX(3) = 0.40D0
PNSTDMX(4) = 0.0

LI88 = .FALSE.
LEXCOM=.FALSE.

LENVG=.FALSE.
ATMSTP=0.010
ENVSTP=0.010

LNEW0=.FALSE.
TRIDT=1.0D-3
TRIDL=8.0D-3
TRITOL=0.0

ATMERR=1.0D-5
ATMD0=1.0D-10
ENVERR=1.0D-5
ATMMAX=0.05
ATMMIN=0.015
ATMBEG=0.015
ENVMAX=0.05
ENVMIN=0.015
ENVBEG=0.015
STOLR0=1.0D-7
IMAX=11
NUSE=7

HTOLER(1,1)=6.0D-4
HTOLER(2,1)=4.5D-4
HTOLER(3,1)=3.0D-4
HTOLER(4,1)=9.0D-4
HTOLER(5,1)=3.0D-4
HTOLER(1,2)=9.0D1
HTOLER(2,2)=5.0D1
HTOLER(3,2)=5.0D1
HTOLER(4,2)=5.0D2
HTOLER(5,2)=2.5E-4

HPTTOL(1)=1.0D-8
HPTTOL(2)=3.0E-2
HPTTOL(3)=0.050
HPTTOL(4)=1.0
HPTTOL(5)=0.01
HPTTOL(6)=0.0010
HPTTOL(7)=0.005
HPTTOL(8)=1.0
HPTTOL(9)=1.0D-1
HPTTOL(10)=0.01
HPTTOL(11)=0.01
HPTTOL(12)=0.0

ATIME(1)=0.005

ATIME(2)=0.0005
ATIME(3)=0.01
ATIME(4)=0.005
ATIME(5)=0.015
ATIME(6)=0.00060
ATIME(7)=0.0060
ATIME(8)=0.02
ATIME(9)=0.04
ATIME(10)=0.02
```

```
ATIME(11)=0.02
LKUT=.FALSE.
NKUT=5
FKUT=2.0
LPTIME=.FALSE.

LROT=.FALSE.
LINSTB=.FALSE.
LJDOT0=.TRUE.
ALFA=1.5
FK=2.46
WALPCZ=0.0
ACFPFT=1.0D-36
ITFP1=5
ITFP2=20
LWNEW=.FALSE.
WNEW=1.2E-7
DTDIF=1.0D-2
ITDIF1=1
ITDIF2=1
DTWIND= 8.0D-2
DJOK=1.0D-4
CZMIN=1.0D-9
FUDGEW=1.0
FUDGEC=1.0
FUDGEO=0.1
FMU=1.0
RCRIT=1000.0
FES=1.0
FGSF=1.0
FSS=1.0
FESC = 1.0D0
FSSC = 1.0D0
FGSFC = 1.0D0
LSMOOTH = .FALSE.

LNEUTR=.TRUE.
LPSEP=.FALSE.
TCUT(1)=6.5
TCUT(2)=6.5
TCUT(3)=6.82
TCUT(4)=7.73
TCUT(5)=7.5
TSCUT=6.3
TENV0=3.0
TENV1=9.0
TGCUT=6.9

LCORE=.FALSE.
FCORE=1.0
MCORE=1

LNEWCP=.FALSE.
ANEWCP='    '

ATMP='REL'
XNEWCP=13

FCORR0=0.8
FCORRI=0.1

LDIAZ=.FALSE.

LVG = .FALSE.
VGZAMS=1.0D10
```

```
 LVGBD= .FALSE.
 VGOM=80
 VGH0=50
 KVG=1
 VGT0=1.5D10
 VGAL=0.008

 LKUTHE=.FALSE.
 OPTOL=1.0D-8

 CMIN = 1.0D-20

 ABSTOL = 1.0D-5
 RELTOL = 1.0D-4
 KEMMAX = 50

 $END

*************************************************************************
*************************************************************************
*************************************************************************
*************************************************************************
*************************************************************************
 ! Default vaules in ().

 ! ******* OPACITY TABLES *******

 TOLLAOL = 10.0        ! (10.0) for LAOL opacities by what facter in density
                       ! and temperature outside table to allow linear extrapolation
 LDELOP = .FALSE.      ! (F) T - Read in a table FDELOP of factors as a function of temp.
                       ! then multiply opacity in model by factor.

 ! ******* EQUATION OF STATE *******

 LOPALE                ! (F) if T then read OPAL eos table
 LOPALE2001            ! (F) if T then use the OPAL2001 eos tables (LOPALE must be T too)
 LOPALD                ! (F) if T and if LOPALE=T then use OPAL derivatives
                       ! otherwise do not. Keep set to F because OPAL
                       ! derivatives are not more accurate and slow down
                       ! execution by a factor of 2
 LMHD = .FALSE.        ! (F) T - use MHD tables for equation of state
 LDH = .FALSE.         ! (F)  T-Add Debye-Huckel correct to Equ. of State.
 ETADH0=-2.0D0         ! (-1.0) assume complete non-degen. electron gass
                       ! for D.H. correction when ETA .LT. ETADH0
 ETADH1=1.0D0          ! (1.0) assume completely degen. electron gas
                       ! for D.H. corr. when ETA .GT. ETADH1; in between
                       ! ramp between two extremes.

! ****** DIFFUSION ******

 LDIFY = .TRUE .       ! (F) T - turn on gravitational settling for He diffusion
 LDIFZ = .TRUE.        ! (F) T - calculate Z diffusion as well as Y diffusion
                       !     requires second set of opacity tables. Assumes
                       !     only Fe fully ionized.
 LTHOUL = .TRUE.       ! (T) T - use new Thoul, Bahcall, and Loeb diffusion
                       ! coef.
 LTHOULFIT = .FALSE.   ! T - use a fit to Thoul, et al diffusion coeff.
                       ! F - use Thoul subroutine to calc diffusion coeff.
 ILAMBDA = 1           ! 1-assign all values of coulomb logs to ln(lambda) = 2.2D0
                       ! 2-assign all values of coulomb logs to ln(lambda)
                       !   from Noerdlinger's formulaI
                       ! 3-assign all values of coulomb logs to ln(lambda)
                       !   from Loeb's formulaI
                       ! 4-calculate table of coulomb logs explicitly
 FGRZ = 1.0            ! (1.0) factor by which the metal diffusion
```

```
                      ! coefficients are multiplied.
 FGRY = 1.0D0         ! (1.0) factor by which the helium diffusion
                      ! coefficients are multiplied.
 DT_GS = 0.1D0        ! Fraction DT_GS of settling timescale which is taken
                      ! as minumum time step in grav. settling of He.
 XMIN = 1.0D-3        ! don't do G.S. calculation if H abundance < XMIN
 YMIN = 1.0D-3        ! don't do G.S. calculation if He abundance < YMIN
 GRTOL = 1.0D-8       ! (1.0D-8) tolerance for diffusion (grav. settling)
 NITER_GS = 10        ! (10) maximum number of iterations

! ****** NONSTANDARD MIXING LENGTH THEORY ********

 LDPREE=.FALSE.       ! (F) invoke Deupree-Varner variable mixing length th.
 MIXKIND=0            ! (0) MIXKIND=0 standard Bohm-Vitense mixing length theory
                      ! MIXKIND=1 use routines in altmix.f and set coef. to
                      ! Bohm-Vitense (see Stothers & Chin ApJ 440, 297)
                      ! MIXKIND=2 use routines in altmix.f and set coef to
                      ! Canuto & Mazitelli formulation
 LHPZ=.TRUE.          ! T-use mix length = alpha * Hp (pressure scale height)
                      ! F-use mix length = alpha * ZDEPTH (depth)
 LQEQ1=.FALSE.        ! T-for MIXKIND=1,2 set Q=1 (common approximation)
                      ! F-for MIXKIND=1,2 set Q=-QDT (correct value)
 MIXVAR=0             ! (0) 0 use standard constant mixing length parameter CMIXLA
 ! MIXVAR > 0 is reserved for nonstandard mixing length formulations. Refer to
 ! altmix.f for full details.
                      ! 1 take ALPHA = HPBETA(2)*ln(P/HPBETA(8))+HPBETA(1)
                      !   HPBETA(1) 0.479
                      !   HPBETA(2) 0.921
                      !   HPBETA(8) is surface pressure (cgs) 1.6E5
                      !   HPBETA(9) is low limit of ALPHA in linear fit
                      !   HPBETA(1) is hi limit of ALPHA in linear fit
                      !   IF LCALS=T then will use CMIXLA as values for HPBETA(1)
                      ! 2 take ALPHA = HPBETA(6) * HWHMOHP where
                      !   HWHMOHP=HPBETA(3)/Hp + HPBETA(4) + HPBETA(5)*Hp
                      !   is taken from a fit to Y.C.'s 3D numerical simulations
                      !   of shallow convection (1995).
                      !   HPBETA(3) = -4.6327E7
                      !   HPBETA(4) = 4.0766
                      !   HPBETA(5) = 3.22866E-8
                      !   HPBETA(6) = 1 but should be adjusted to fit sun.
                      !   IF LCALS=T then use CMIXLA as values fo HPBETA(4)
                      ! 3 use a quartic fit in Hp to Y.C.'s 1995 simulations
                      ! 4 see altmix.f for other formulations
 HPBETA(1)=0.0D0      ! (0) for MIXVAR.NE.0
 HPBETA(2)=0.0D0
 HPBETA(3)=0.0D0
 HPBETA(4)=0.0D0
 HPBETA(5)=0.0D0
 HPBETA(6)=0.0D0
 HPBETA(7)=0.0D0
 HPBETA(8)=0.0D0
 HPBETA(9)=0.0D0
 HPBETA(10)=0.0D0

 ! ********** NONSTANDARD MIXING **********

 LNSTDMX=.FALSE.      ! force nonstandard mixing controlled by PNSTDMX
 PNSTDMX(1) = 1.0D0   ! = 1 then mix from interior to mass fraction PNSTDMX(2)
                      ! = 2 then mix from interior to mass fraction PNSTDMX(2)
                      !   once when the age is equal to PNSTDMX(3). PNSTDMX(4) is used
                      !   to tell if mixing has been done.  It should be set to 0.0.
                      ! = 3 then mix from mass fraction PMSTDMX(2) to PNSTDMX(3)
 PNSTDMX(2) = 0.1
 PNSTDMX(3) = 0.1
 PNSTDMX(4) = 0.0
```

```
 ! ****** OVERSHOOT, SEMICONVECTION ******

 LADOV = .FALSE.       ! (F) T - extend abiabatic temp grad below base
                       ! of c.z. (envelope) by ALPHAE pressure scale heights
                       ! Must set LOVSTE to T
 LSEMIC = .FALSE.      ! (F) T - turn on semiconvection routine
 DPENV=1.0             ! (1.0) artificially mix from surface c.z.
                       ! to 1-Mov/Mstar = DPENV
 LOVSTC=.FALSE.        ! (F) F - do not overshoot above core C.Z.; T - do
 ALPHAC=0.0            ! (0.0) Extent of core C.Z. overshoot
                       ! layer (Pscale height)
 LOVSTE=.FALSE.        ! (F) F - do not overshoot in envelope C.Z.; T - do
 ALPHAE=0.5            ! (0.0) Extent of envelope C.Z. overshoot layer
 LOVSTM=.FALSE.        ! (F) F - do not overshoot in middle C.Z.s; T - do
 ALPHAM=0.0            ! (0.0) Extent of middle C.Z. overshoot layers

! ******* NUCLEAR BURNING *********

 LNEWE = .TRUE.        ! (F) T- use Bahcall (ENGEB) rates to calculate nuclear
                       ! luminosities; F - use Prather (ENGE) formulism to
                       ! calc nuclear luminosities
 LSNU=.FALSE.          ! (F) if LNEWE=T calculate neutrino fluxes and
                       ! print them out to ITRACK,ISHORT,IMODPT
 LNEWM = .TRUE.        ! (F) if T then use NEWMIX otherwise use old MIX;
                       ! require LNEWE=T
 LI88 = .FALSE.        ! If T use 1988 Li burning rates otherwise use
 LEXCOM=.FALSE.        ! (F) track abundance of H2, L16, L17, and Be9

 ! ******** MISC ********

 LATMTAB=.FALSE.       ! (F) if T then interpolate Kurucz model atmoshpere
                       ! table (FATM)
 LENVG=.FALSE.         ! calc. env.&atm. for converged model at Teff, L of
                       ! model
 ATMSTP=0.30           ! (0.5) [0.01] integ. step for atm. calc. also
                       ! for PULSE.ATMOS
 ENVSTP=0.30           ! (0.5) [0.01] integ. step for env. calc. also
                       ! for PULSE.ENVEL

 KTTAU=0               ! if 0 then use Eddington T Tau if 1 use Krishna-Swamy

 LCORE=.FALSE.         ! (F) T - after initial model is read in
                       ! extend inner shell (doesn't work very well)
 FCORE=1.0             ! (1.0)  inner most shell's mass is HS(1)/FCORE
 MCORE=1               ! (1) between HS(1) and HS(1)/FCORE put MCORE shells

 LNEWCP=.FALSE.        ! rescale abund. of ANEWCP
 ANEWCP='   '          ! element whose abundance is to be rescaled. One
                       ! of:HE3,C12,C13,N14,N15,O16,O17,O18,H2,LI6,LI7,BE9.
 ATMP='REL'            ! or 'ABS' relative to log H = 12 or absolute abund.
 XNEWCP=13             ! desired rescaled abund. of ANEWCP


 !            ******  CONVERGENCE, SHELL RESOLUTION  ********

 ! ****** ITERATION CONTROL ********

 LNEWS = .TRUE.        ! if T then retain last model's entropy term;
                       ! speeds up evolution
 NITER1=2              ! (2) max. no. of iterations for first corrections
 NITER2=40             ! (20) max. no. of iter. for second set of corrections
 NITER3=10             ! (0) if LNEWE=T then set a third level of iteration
                       ! to NITER3=10
 NITER4 =  0           ! (0) for solar models and LNEWE=T set a forth level
```

```
                      ! of iteration NITER4=10
FCORR0=0.8            ! (0.8) initial correction factor
FCORRI=0.1            ! (0.1) increment to correction factor after
                      ! each iteration

LNEW0=.FALSE.         ! (F) calc. new envel. triangle for each model
TRIDT=1.0D-2          ! (0.01) [0.001] length of temp edge of env. tri.
TRIDL=8.0D-2          ! (0.08) [0.008] length of lum. edge of env. tri.
TRITOL=0.0            ! (0.0) fraction model can move outside tri. before
                      ! flipping

! ******* ENVELOPE AND ATMOPSHERE RESOLUTION

ATMERR=3.0D-4         ! (3.0D-4) max error in atmos. integ.
ATMMAX=0.1            ! (1.0D-1) max integ. step in log Tau
ATMMIN=0.02           ! (0.10) min integ. step in log Tau
ATMBEG=0.1            ! (0.10) inital integ. step in log Tau
ATMD0=1.0D-10         ! (1.0D-10) density zero-point
ENVERR=3.0D-4         ! (3.0D-4) max error in envel. integ.
ENVMAX=0.50           ! (0.5) max integ. step in log P for envel. integ.
ENVMIN=0.02           ! (0.25) min integ. step in log P for envel. integ.
ENVBEG=0.10           ! (0.35) initial step in log P for envel. integ.
STOLR0=1.0D-3         ! (1.0D-3) tol. for fitting mass integ./extrap.
IMAX=11               ! (11) no. of times derivatives evaluated across
                      ! interval
NUSE=7                ! (7) no. of prev. guesses of the end step

! ******* INTERIOR RESOLUTION *******

HTOLER(1,1)=6.0D-5    ! (6.0D-5) conv. criterion for delta(log P)
HTOLER(2,1)=4.5D-5    ! (4.5D-5) conv. criterion for delta(log T)
HTOLER(3,1)=3.0D-3    ! (3.0D-5) conv. criterion for delta(L/Lsun)
HTOLER(4,1)=9.0D-5    ! (9.0D-5) conv criterion He burning
HTOLER(5,1)=3.0D-5    ! (3.0D-5) conv. tol. of diff. equs. for log P and log R
HTOLER(1,2)=9.0D-1    ! (9.0D-1) max allowed correction for delta(log P)
HTOLER(2,2)=5.0D-1    ! (5.0D-1) max allowed correction for delta(log T)
HTOLER(3,2)=5.0D-1    ! (5.0D-1) max allowed correction for delta(L/Lsun)
HTOLER(4,2)=2.0D0     ! (2.0)
HTOLER(5,2)=2.5E-6    ! (2.5D-6) conv. tol. on RHS of T and L equs.

HPTTOL(1)=1.0D-8      ! (1.0D-8) min change in log M between Henyey grid pts
HPTTOL(2)=8.0E-2      ! (8.0D-2) max change in log M between Henyey grid pts
HPTTOL(3)=5.0E-2      ! (5.0D-2) DELTA X between 2 grid pts. required to
                      ! flag pts.
HPTTOL(4)=1.0         ! (1.0) DELTA Y between 2 grid pts to flag
                      ! (1.0 - no flag)
HPTTOL(5)=0.050       ! (0.05) max del log P in surf. c.z./above fine mesh
HPTTOL(6)=0.020       ! (0.02) max change in log(L/Lsun) between
                      ! Henyey grid pts.
HPTTOL(7)=1.0         ! (1.0) max discontinuity in X between 2 grid pts.
HPTTOL(8)=1.0         ! (1.0) max discontinuity in Z between 2 grid pts.
HPTTOL(9)=1.0D-1      ! (0.1) max del log Omega otherwise point is flagged
HPTTOL(10)=0.050      ! (0.05) max del log P  in fine mesh zone
HPTTOL(11)=0.050      ! (0.05) max del log P below surf. c.z.
                      ! and below fine mesh zone
HPTTOL(12)=0.0        ! (0.0) del log P fine mesh extends below
                      ! over shoot and above base surf. c.z.

! ******* TIME STEPPING ********

ATIME(1)=0.001        ! (0.001) min. central X before H burning
                      ! defined in shell
ATIME(2)=0.02         ! (0.02) max. change in X for core H burning (m.s.)
ATIME(3)=0.5          ! (0.5) max. fractional change in central X (turn off)
ATIME(4)=0.02         ! (0.02) max. change in Y in central regions
```

```
 ATIME(5)=0.03          ! (0.03) max. fractional change in Y at center
 ATIME(6)=0.0015        ! (0.0015) max mass allowed to burn in shell
 ATIME(7)=0.1           ! (0.1) max depeletion in X at midpoint of H-shell
 ATIME(8)=0.005          ! (0.02) max change in P from one model
                        ! to next (LPTIME=T)
 ATIME(9)=0.01          ! (0.04) max change in T from one model
                        ! to next (LPTIME=T)
 ATIME(10)=0.005         ! (0.02) max change in R from one model
                        ! to next (LPTIME=T)
 ATIME(11)=0.005         ! (0.02) max change in L from one model
                        ! to next (LPTIME=T)
 LKUT=.FALSE.           ! (F) use time step cutting
 NKUT=5                 ! (5) max. no. of time step cutting attempts
 FKUT=2.0               ! (2.0) time step cutting factor
 LPTIME=.FALSE.         ! (F)use Pachenski time step criterion


 !            ******* PARAMETERS FOR ROTATION ********

 LROT=.FALSE.           ! global flag F-nonrotating model, T-rotating model
 LINSTB=.FALSE.         ! calculate hydrodynamic rot. induced instabilites
 LJDOT0=.TRUE.          ! use Belcher-MacGregor model wind
 ALFA=1.5               ! (1.5) wind law constant (exponent)
 FK=2.46                ! (2.46) wind law constant factor
 WALPCZ=0.0             ! enforce rotation law: omega = r**walpcz in C.Z.
 ACFPFT=1.0D-36         ! relative accur. of equipotential surface determination
 ITFP1=5                ! no. of iter. between calc. ETA2 and R0
 ITFP2=20               ! no. of iter. to calc. R0 and HR
 LWNEW=F                ! enforce rigid rotation in model
 WNEW=1.2E-7            ! Initial angular velocity of model
 DTDIF=1.0D-2           ! Maximum change in omega at any point for
                        ! one diffusion  time step
 ITDIF1=1.0
 ITDIF2=1.0
 DTWIND= 8.0D-2         ! Maximum change in omega at any point
                        ! for one model timestep
 DJOK=1.0D-4            ! Tolerance for diffusion equation
 CZMIN=1.0D-9           ! min. depth of c.z. for angular momentum loss.
 FUDGEW=1.0             ! Diffusion coeff for angular momentum are
                        ! multiplied by FUDGEW
 FUDGEC=1.0             ! Diffusion coeff for composition are multiplied
                        ! by FUDGEC
 FUDGEO=0.1             ! Multiply diff. coeff at base of C.Z. by FUDGEO
 FMU=1.0                ! Sensitivity of ES and GSF instabilities to mu
                        ! gradients are prop to FMU
 RCRIT=1000.0           ! critical Reynolds number at which point shear
                        ! instabilites set in
 FES=1.0                ! Factors by which ES velocity estimates are multiplied
 FGSF=1.0               ! Factors by which GSF velocity estimates are multiplied
 FSS=1.0                ! Factors by which shear velocity estimates are
                        ! multiplied
 FESC = 1.0D0           ! (1.0) Diffusion coefficient for Eddington-Sweet
                        ! instability
 FSSC = 1.0D0           ! (1.0) Diffusion coefficient for secular-shear
                        ! instability
 FGSFC = 1.0D0          ! (1.0) Diffusion coefficient for GSF instability
 LSMOOTH = .FALSE.      ! (F) not implimented

! ******* VARYING COSMOLOGICAL G *********

 LVG = .FALSE.          ! (F) T-use varying G; F-standard constant G
 VGZAMS=1.0D10          ! age of Universe when star is on ZAMS
 LVGBD= .FALSE.         ! T-use Brans-Dicke formulation.  Must specify
                        ! VGH0, VGOM, in addtion to VGZAMS
                        ! F-use other formulation depending on KVG.
```

```
VGOM=80               ! Brans-Dicke omega
VGH0=50               ! Hubble's constant in Km/s-Mpc
KVG=1                 ! (1) 1-use power-law formualtion.
                      ! Mustspecify VGAL, VGT0, in addition to VGZAMS
VGT0=1.5D10           ! Current age of the Universe in years
VGAL=0.008            ! exponent of power law formulation

!               ******* DON'T TOUCH PHYSICS ******

LNEUTR=.TRUE.         ! (T) turn on/off leptonic neutrino losses (leave on)
LNULOS1=.FALSE.       ! (F) Use new pair, photo, plasma neutrino loss calculation
                      !  (LNEUTR must be TRUE)
LNULOS2=.FALSE.       ! (F) Calculate neutrino losses for He,C,O Bremsstrahlung
                      !  (LNEUTR must be TRUE and LNULOS1 must be TRUE)
LPSEP=.FALSE.         ! (F) use old energy generation network (leave it off)
TCUT(1)=6.5           ! (6.5) low temp cutoff for all nuclear energy generation
TCUT(2)=6.5           ! (6.5) low temp cutoff for pp chains
TCUT(3)=6.82          ! (6.82) low temp cutoff for CNO cycle
TCUT(4)=7.73          ! (7.73) low temp cutoff for all He burning
TCUT(5)=7.5           ! (7.5) low temp cutoff for leptonic neutrino emission
TSCUT=6.3             ! (6.3) high temp cutoff for Saha equation
TENV0=3.0             ! (3.0) minimum  envelope temp
TENV1=9.0             ! (9.0) maximum envelope temp
TGCUT=6.9             ! (6.9) do not use mix. theory to get T grad. above
                      ! this temp
LDIAZ=.FALSE.         ! (F) use Gust.-Bell model atmos. to get i.c. for
                      ! envel. integr.
LKUTHE=F              ! invoke modifications for He flash calculation
IHPOINT=1             ! call HPOINT every IPOINT models
LINHPNT=F             ! (F) if T then use linear interpolation in HPOINT
                      ! otherwise use oscilatory spline.  The latter
                      ! introduces oscillations near tip of GB
OPTOL=1.0D-8          ! (1.0D-8) if compos. is "near" opac. table grid
                      ! do not interp.
CMIN = 1.0D-20        ! constant used to check against division by zero
                      ! in CHECKC and NKEMCOM,
ABSTOL = 1.0D-5       ! Absolute tolerance in NKEMCOM
RELTOL = 1.0D-4       ! Relative tolerance in NKEMCOM
KEMMAX = 50           ! Maximum number of iterations in NKEMCOM
```

Using YREC

YREC has two different mechanisms for defining multiple runs. The original kind card loop is defined within a single NAMELIST pair (PHYSICS and CONTROL). One can define up to 50 different runs. All the runs use the same set of parameters that are defined in the NAMELIST pair. Greater flexibility is obtained by using the NAMELIST loop. Here up to 100 different NAMELIST pairs can be defined (this number can be changed by modifying main.f). YREC will execute all the runs defined by each NAMELIST pair. The NAMELIST loop permits one to define runs with completely different parameters. The NAMELIST loop was implemented after the kind card loop and indeed there is a great deal of redundancy in the two looping structures. The primary feature of the NAMELIST loop is that each run or runs of the NAMELIST loop can have its own set of parameters, where as the runs in the kind card loop must all use the same set of parameters. The bottom line, is that the NAMELIST loop offers greater control but to maintain backward compatibility, the kind card loop has been retained. Often, you will be able to accomplish the same set of runs using either the NAMELIST loop or the kind card loop, (or even a combination of both).

NAMELIST Loop

With version 7 of YREC, it is now possible to carry out runs with different or distinct NAMELISTs pairs (*.nml1 and *.nml2). When YREC is run, it looks for the NAMELIST file ($NMLCONTROL) named "yrec7.batch" in the same directory as YREC and reads it. The NAMELIST loop was implemented to facilitate computing a variety of different stellar models in one run of YREC. In the following example of a NMLCONTROL NAMELIST, i.e., yrec7.batch file:

```
$NMLCONTROL

NUMNML=3

FNML1(1)='E:\dguenther\research\MOST\nml\sbd06.nml1'
FNML2(1)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

FNML1(2)='E:\dguenther\research\MOST\nml\sbd07.nml1'
FNML2(2)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

FNML1(3)='E:\dguenther\research\MOST\nml\sbd08.nml1'
FNML2(3)='E:\dguenther\research\MOST\nml\subdwarf.nml2'

$END
```

NUMNML specifies the number of NAMELIST pairs to read. The character arrays (CHAR*256), FNML1(I) and FNML(2), define the files containing the two NAMELISTs (CONTROL and PHYSICS), for the Ith NAMELIST run. Up to 100 NAMELIST pairs can be defined (this can be changed by modifying main.f). By using the NAMELIST loop rather than the "kind card" loop, one can specify unique parameters for each run.

Kind Card Loop

For each pair of NAMELISTs (CONTROL and PHYSICS) that YREC reads, YREC can carry out up to fifty different runs, one after the other. Each run can be either a rescaling run where some of the physical characteristics, e.g., mass, of the initial model are rescaled, an evolutionary run where time evolving sequences of models are constructed, or a rescaling and evolving run where both rescaling and evolution take place at the same time, this latter option must be used when rescaling a pre-main-sequence model.

Consider for example, the following five runs defined in a $CONTROL NAMELIST:

We will execute 5 kind card loop runs.

```
NUMRUN=5                ! number of runs; maximum is 50
```

The first run will read in (LFIRST(l)=.TRUE.) a ZAMS model and rescale (KINDRN(1)=2) its mass to 0.95 M_s (RSCLM(l)=0.95), rescale its hydrogen mass fraction to 0.701 (RSCLX(1)=0.701), and not rescale anything else (negative values mean do not rescale). The rescaled model will be run through the structure equations twice to relax the model (NMODL5(1)=2). The mixing length parameter (CMIXL) must be set for each run. See section on solar models for recommended values.

```
! rescaling run
KINDRN(1)=2             ! rescaling run
LFIRST(l)=.TRUE.        ! read in initial model from a file
RSCLM(l)= 0.95          ! rescale initial mass of model to 0.95Msun
RSCLX(1)=0.701          ! rescale X throughout model to 0.701
```

```
 RSCLZ(1)=-0.0194        ! do not rescale Z
 RSCLCM(1)=-1.0          ! do not rescale core mass
 NMODL5(1)=2             ! relax (entropy term=0) model twice
 XENV0A(1)=0.701         ! specify initial envelope X
 ZENV0A(1)=0.0194        ! specify initial envelope Z
 CMIXLA(1)= 1.19         ! specify mixing length parameter
 LSENV0A(1) = .FALSE.    ! do not reposition envelope fit point
 SENV0A(1) = 2.0D-4      ! shift envelope fit point to this mass fraction
```

Once the first run is completed and the results output, the second run will begin. The second run will take the rescaled model from the first run (LFIRST(2)=.FALSE.) and evolve it (KINDRN(2)=1) to 5 Gyrs (ENDAGE(2)=5.0E9). The time-steps are forced to be equally spaced at 0.25 Gyrs (SETDT(2)=0.25E9). The run will take 20 models. The number of models must be set to be equal to or greater than this (NMODL5(2)= 50), otherwise the run will not get to 5 Gyrs.

```
 ! evolving run
 KINDRN(2)=1              ! evolutionary run
 LFIRST(2)=.FALSE.        ! use last model from previous run
 ENDAGE(2)=5.0E9          ! evolve until age is 5Gyr
 SETDT(2)= 0.25E9         ! use fixed time-steps of 0.25Gyr
 NMODL5(2)= 50            ! maximum number of models in this run (only 20 required)
 XENV0A(2)=0.701          ! initial X
 ZENV0A(2)=0.0194         ! initial Z
 CMIXLA(2)= 1.19          ! mixing length parameter
```

The second run will output its results. The third run will read in the ZAMS model again and rescale its mass to 0.95 M_s and its X to 0.701 (the same as the first rescaling run). It cannot access the rescaled model output from the first run, only the output from the immediately preceding run, which is why we need to read in the starting model again.

```
! rescaling run
 KINDRN(3)=2             ! rescaling run
 LFIRST(3)=.TRUE.        ! read in initial model from a file
 RSCLM(3)= 0.95          ! rescale initial mass of model to 0.95Msun
 RSCLX(3)=0.701          ! rescale X throughout model to 0.701
 RSCLZ(3)=-0.0194        ! do not rescale Z
 RSCLCM(3)=-1.0          ! do not rescale core mass
 NMODL5(3)=2             ! relax (entropy term=0) model twice
 XENV0A(3)=0.701         ! specify initial X
 ZENV0A(3)=0.0194        ! specify initial Z
 CMIXLA(3)= 1.19         ! specify mixing length parameter
 LSENV0A(3) = .FALSE.    ! do not reposition envelope fit point
 SENV0A(3) = 2.0D-4      ! shift envelope fit point to this mass fraction
```

We will now rescale an already rescaled model. The fourth run will take the previously rescaled ZAMS model and rescale its mass to 0.90 M_s. It is not possible to rescale by large differences. Although, YREC will happily carry out the rescaling, when it comes time to evolve the rescaled model, the rescaled model may not be accurate enough and YREC will not be able to converge to an evolved model. As the X has already been rescaled in the third run it can be left alone.

```
 ! rescaling run
 KINDRN(4)=2             ! rescaling run
 LFIRST(4)=.FALSE.       ! use last model of previous run
 RSCLM(4)= 0.90          ! rescale initial mass of model to 0.9Msun
 RSCLX(4)=-0.701         ! do not rescale X
 RSCLZ(4)=-0.0194        ! do not rescale Z
 RSCLCM(4)=-1.0          ! do not rescale core mass
 NMODL5(4)=2             ! relax (entropy term=0) model twice
 XENV0A(4)=0.701         ! specify initial X
 ZENV0A(4)=0.0194        ! specify initial Z
 CMIXLA(4)= 1.19         ! specify mixing length parameter
```

```
LSENV0A(4) = .FALSE.     ! do not reposition envelope fit point
SENV0A(4) = 2.0D-4       ! shift envelope fit point to this mass fraction
```

The fifth run will take the final rescaled ZAMS model (0.90 M_s) and evolve it for 500 models (NMODL5(5)= 500). Time-steps are not explicitly specified hence will be chosen by the program (SETDT(5)= -0.25E9).

```
! evolving run
KINDRN(5)=1              ! evolutionary run
LFIRST(5)=.FALSE.        ! use last model of previous run
ENDAGE(5)=-5.0E9         ! no final age is specified
SETDT(5)= -0.25E9        ! YREC will adjust time-step automatically
NMODL5(5)= 500           ! calculate 500 models
XENV0A(5)=0.701          ! initial X
ZENV0A(5)=0.0194         ! initial Z
CMIXLA(5)= 1.19          ! mixing length parameter
```

YREC will carry out the runs sequentially. Notice that the result of preceding run (a rescaling run in the example) can be used as the input model for another run (either a rescaling or evolutionary run.)

Before running YREC you must indicate the number of runs you want and the nature of each run. This is the first parameter listed in the $CONTROL NAMELIST file. Other than that, typically you will only need to specify the mixing length and composition.

YREC will use the same parameters, e.g., opacity tables, Z values, etc., for every kind card run. To specify different parameters you must define unique NAMELIST pairs for each run, and use the NAMELIST loop to carry out the runs.

The only way to rescale a pre-main-sequence model is to allow it to evolve immediately afterward, otherwise the model solution will not converge. This is done by setting KINDRN(I)=3. With this setting YREC will rescale the model, and then carry out a tiny evolutionary step. The evolutionary step allows the entropy term to relax to an appropriate value, which is necessary on the pre-main-sequence because the star is collapsing very rapidly.

Check list

Even for a simple stellar evolutionary run, there are many things to remember to do when setting up the NAMELIST files in preparation for a run. Unfortunately, because there are so many parameters to consider that it is rare to get everything right the first time. It is therefore important to check your resultant models very carefully, to make sure you have not made any obvious mistakes. If, for example, you set the composition to Z=0.002 but do not remember to set the opacity file or the equation of state files to this value, YREC will simply go ahead and carry out the calculation using inconsistent opacity and equation of state tables. No error message will be printed, and you won't discover your error until you compare your models to other models.

We recommend setting up a reference pair of NAMELIST (CONTROL and PHYSICS) files that you know are correct. Then for other runs, start with the reference pair and adjust only those parameters that need changing.

Here is a short list of the items you need to assemble to produce a *simple* run. After reading this, have someone help set up a run for you. Then re-read this section, it will make a lot more sense after you have actually completed a run.

1. Locate a starting model of the same mass as the star you want to evolve. Many such models have been archived and are readily available. If no such model exists then you will have to plan on scaling down (not up) a model to the mass you want (using a rescaling run). If you need to calculate pre-main-sequence models you will need to obtain a pre-m.s. starting model. Initial pre-m.s. models are constructed using a Lane-Emden equation solver. You need to define the directory and file name of the starting file in the CONTROL NAMELIST (parameter FFIRST).

2. Locate the opacity files you need. As of this writing we recommend using the OPAL95 (LOPAL95=.TRUE.) and Alexander tables (LALEX95=.TRUE.), the former cover the higher temperature regime and the latter the lower temperature regime. In the CONTROL NAMELIST define the directory and file names, if they are not already set, and specify the Z of your models. Note that if you are doing Horizontal Branch evolution then you need to set LPUREZ=.TRUE. and specify the location of the pure Z table. In addition, you need to specify a second Z value if you are following Z diffusion.

3. Next go through the kind card parameters, the parameters that define the run as evolving or rescaling, and set these parameters. You have to be particularly careful in setting the composition. There is some redundancy in these settings as you must set X and Z twice for each run that involves rescaling: in the rescaling parameters (RSCLX, RSCLY) and in the initial parameters (XENV0A, ZENV0A).

4. If you are using the OPAL equation of state tables than you need to locate a table with the appropriate Z. A separate program is available to produce OPAL equation of state tables of different Z. The location of the table and its Z value must be specified in the CONTROL NAMELIST. The flag that determines whether you are using the OPAL equation of state tables or not is located in the PHYSICS NAMELIST (LOPALE). If you do not use the OPAL equation of state tables, then YREC will use a built-in analytic form, for which you do not need to specify the Z.

5. Check that LRSH=.TRUE. and other output parameters are set to .FALSE. unless you really want to generate a huge amount of output on your run. Although, when getting started it is recommended that you turn on all the output while keeping the number of models in your runs to less than 100. You should examine the output very carefully to make sure everything is behaving as expected. Then when you are confident that everything is working correctly, turn off all the output (LSCRIB=.FALSE., LRWSH=.TRUE.) except the track output (LTRACK=.TRUE.).

6. If you are going to use the models for pulsation make sure the LPULSE flag is set to .TRUE.. Also, make sure the shell resolution variables are set to produce about 600 shells each in the interior, envelope, and atmosphere (see detailed description of PHYSICS NAMELIST variables).

7. Check that you are using an Eddington gray atmosphere KTTAU=0, in the PHYSICS NAMELIST if you are doing stellar models, and Krishna-Swamy atmosphere KTTAU=1 if you are doing solar models.

8. Check that the diffusion variables, and the overshoot, and semi-convection variables are set correctly. If you are doing diffusion in Z, you also need to specify a second Z opacity value for both the low temperature and high temperature opacities.

9. Set the maximum number of models and/or the final age of the model appropriately.

10. Run the program, and wait for a few minutes for all the files to be read in. Most errors are a consequence of mistyped file names, and YREC will fail within a few minutes of starting up. Error messages are printed out in the FSHORT file.

To a novice user's dismay, there are several hundred different parameters that need to be learned if you want to be able to run the code independent of an experienced user. You will slowly master these parameters when things go wrong and you need to adjust one particular parameter in order to make the code go.

In my experience training students to use the code, one can expect it will take about a week to learn how to produce a simple run for the Sun. A good student will probably spend a month working with the code before they are comfortable producing models of stars. And it will typically take from one to two years of experience with the code before you will feel confident enough to make competent modifications to the code. As mentioned in the introduction, the code literally embodies nearly everything we know about stellar structure and evolution, and to master the entire code requires years of study.

Calibrated Solar Models

YREC is capable of automatically calculating a tuned solar model. The user specifies how close they want the radius and luminosity of the evolved model to match the observed radius and luminosity of the sun and YREC does the rest. YREC does this by evolving a model, testing how close it comes to hitting the Sun's luminosity and $T_{\rm eff}$ at the Sun's age, then trying again with appropriate adjustments to the mixing length parameter (which predominately controls the radius) and the helium abundance (which predominately controls the luminosity) until it finally produces a hit.

It is from a calibrated solar model that we obtain the helium abundance of the Sun (it cannot be directly observed in the photosphere). In addition, the calibrated mixing length parameter, for the specific physics options chosen, is used when modeling other stars.

The luminosity and radius (hence $T_{\rm eff}$) of the Sun are hard coded in to YREC (setups.f). The age is not, hence, must be specified.

Consider the following lines taken from the first lines of a typical $CONTROL NAMELIST used to construct a solar model

```
$CONTROL

DESCRIP(1)='Standard Solar Model with D.H.'
DESCRIP(2)='LAOL=LIVOP=LKUMOL=T KTTAU=1 LDH=T NITER3=NITER4=10'

LCALS = .TRUE.
TOLL = 1.0D-6
TOLR = 1.0D-7
LINCLUDEZX = .FALSE.
TOLZX = 1.0D-4
ZOVERX = 0.0244D0

NUMRUN=50

KINDRN(1)=2
LFIRST(1)=.TRUE.
RSCLM(1)= -1.0D0
RSCLCM(1)=-1.0
RSCLX(1)=0.710638475D0
RSCLZ(1)=0.01880D0
RSCLZC(1)=-0.0001D0
RSCLZM1(1)=-0.2D0
RSCLZM2(1)=-0.3D0
NMODLS(1)= 1
```

```
XENV0A(1)=0.710638475D0
ZENV0A(1)=0.0188D0
CMIXLA(1)= 1.9D0

KINDRN(2)=1
LFIRST(2)=.FALSE.
ENDAGE(2)=4.50D9
SETDT(2)= 0.09D9
NMODLS(2)= 55
XENV0A(2)=0.710638475D0
ZENV0A(2)=0.0188D0
CMIXLA(2)= 1.9D0
```

The parameter LCALS=.TRUE. tells YREC that you want to calculate a tuned solar model. TOLL and TOLR and the tolerances for the luminosity and radius, respectively. Specifically, YREC will assume it has a tuned solar model if $L/L_{\odot} \leq$TOLL and $R/R_{\odot} \leq$TOLR.

You must specify some of the parameters for the first two runs. YREC will automatically fill in the parameters for any additional runs it needs to produce a tuned solar model. The first run must be a rescaling run (KINDRN=2), even if you do not need to rescale your initial model. The second run must be an evolution run (KINDRN=1). Here you must also specify the age of the sun (4.49±0.03Gyr). It is recommended for stability that you set SETDT to a fixed time-step. As shown in the example above, YREC will evolve an initial model in 50 equal time-steps of 0.09Gyr each to an age of 4.5Gyr. NMODLS has been set accordingly. (YREC will stop each run when NMODLS have been calculated or when the age is reached, whichever happens first.)

YREC will evolve an initial rescaled model as specified by you in the first two runs. It will then set up, automatically, a third run to rescale the initial model, and a fourth run to evolve the rescaled initial model. YREC adjusts both the helium abundance and the mixing length parameter to tune the luminosity and radius, that is, the final solar model will have different *X*, *Y*, and *Z* then specified in the NAMELIST parameters, and it will have a different mixing length parameter. YREC will improve the accuracy of radius and luminosity of the model by approximately a factor of ten through each iteration of rescaling and evolving.

You can also tune to a specific Z/X by setting LINCLUDEZX=.TRUE. Otherwise, the code will use the initial Z and specified by RSCLZ. Note that with diffusion this initial Z and X will not be the final evolved Z and X. When LINCLUDEZX=.TRUE. the code will iterate on Z (along with Y and alpha) to get a model that at the age of the Sun gives Z/X=ZOVERX to an accuracy of TOLZX.

A standard solar model constructed using, Krishna-Swamy atmosphere, Alexander and OPAL95 opacities and the OPAL equation of state yields the following values, when Z/X= 0.0245:

SSM: 4.5Gyr, X=0.719567, Z=0.017630, alpha=1.8621

SSM with He diffusion: 4.5Gyr, X=0.718010, X_{env}=0.735125, Z=0.018011, α=1.96396.

SSM with He and Z diffusion: 4.5Gyr, X=0.70576, X_{env}=0.73572, Z=0.020, Z_{env}=0.01803, α=2.079.

SSM with He and Z diffusion: 4.55Gyr, X=0.70623, X_{env}=0.73798, Z=0.020, Z_{env}=0.01802, α=2.087.

Note α will be lower by ~0.23 when using the Eddington gray atmosphere. We recommend for stars setting $\alpha = 1.6$ for models without diffusion and setting $\alpha = 1.8$ for models with diffusion.

Expect different values when using different opacities, different equation of state, different diffusion options, and different composition.

Calibrated Stellar Models

An option exists to have YREC automatically calculate a stellar model of specified mass, radius and luminosity. To activate this option set LCALST=.TRUE. in the $CONTROL NAMELIST. The luminosity, in solar units, is specified with XLS and its tolerance, in solar units, is XLSTOL. You can specify either the radius desired, in solar units, using SR, or the effective temperature, in Kelvin, using STEFF. Set LTEFF=.FALSE. if you are specifying the radius, and set LTEFF=.TRUE. if you are specifying the effective temperature. You can have YREC adjust either the helium abundance or the mixing length parameter to produce a tuned model (LADJX). As with the solar model calibration option you must specify the first run as a scaling run and the second run as an evolution run. YREC will automatically add runs as necessary to converge to the desired model. Normally you cannot specify equal time-steps for models beyond the main sequence, hence, you should set NMODLS large enough to get to the point of evolution you want.

When you run YREC with this option, YREC will start with an initial run as specified by your scaling and evolution run. As soon as the star just passes the specified radius it will note the luminosity of the star at that point. It will then calculate another run with the helium mass fraction (LADJX=.TRUE.) increased by 0.01 or the mixing length parameter increased by 0.1 (LADJX=.FALSE.), again stopping just after the radius of the star passes the specified radius. The results of these two runs will be used to determine a guess for the helium mass fraction or mixing length parameter, respectively. The iterative procedure will repeat until a model passes through the specified luminosity and radius. When an evolution track passes through the specified luminosity and radius, YREC calculates the exact age at which the stellar model passes through the radius-luminosity target and then sets up a final run which re-runs the last run except that the run is set to stop at the precise age at which the model hits the target radius-luminosity.

Due to nonlinear effects and numerical instabilities caused by the unequal size time-steps YREC cannot hit a target radius and luminosity with the same accuracy as it can a solar model. XLSTOL should not be set to less than 0.5% the luminosity of the target model otherwise YREC will not be able to hit the target.

A Note on Giant Branch Evolution

The PHYSICS NAMELIST parameters must be carefully set to ensure convergence all the way up the giant branch, and to ensure that the evolution is accurate. Critical attention must be paid to the time step parameters ATIME(6) and ATIME(7). If set too coarsely the evolution will not be accurate (even though the models are converged). Also, it is critical to ensure that there is sufficient number of shells in the H burning (shell) region, otherwise features, such as the zig-zag, will be computed incorrectly. We recommend starting with the following parameters and then adjusting from this set as suits your needs. The luminosity convergence criterion, HTOLER(3,1)=3.0D-3, needs to be relaxed with higher mass stars since it is an absolute measure.

DBG does not believe that third and forth iterations NITER3 and NITER4 provide numerically stable and correct solutions. DBG sets NITER3 and NITER4 = 0 on the giant branch.

```
HTOLER(1,1)=6.0D-4
HTOLER(2,1)=4.5D-4
HTOLER(3,1)=3.0D-4
HTOLER(4,1)=9.0D-4
HTOLER(5,1)=3.0D-4
HTOLER(1,2)=9.0D1
HTOLER(2,2)=5.0D1
HTOLER(3,2)=5.0D1
HTOLER(4,2)=5.0D2
HTOLER(5,2)=2.5E-4
```

```
HPTTOL(1)=1.0D-8
HPTTOL(2)=3.0E-2
HPTTOL(3)=0.050
HPTTOL(4)=1.0
HPTTOL(5)=0.01
HPTTOL(6)=0.0010
HPTTOL(7)=0.005
HPTTOL(8)=1.0
HPTTOL(9)=1.0D-1
HPTTOL(10)=0.01
HPTTOL(11)=0.01
HPTTOL(12)=0.0

ATIME(1)=0.005

ATIME(2)=0.0005
ATIME(3)=0.01
ATIME(4)=0.005
ATIME(5)=0.015
ATIME(6)=0.00060
ATIME(7)=0.0060
ATIME(8)=0.02
ATIME(9)=0.04
ATIME(10)=0.02
ATIME(11)=0.02
```

Mass Loss and Gain

Section to be provided. Yrec7d contains subroutines to calculate mass loss and gain.

Rescaling initial models

Over the years we have produced a number of ZAMS models of different mass and composition. It is recommended that you use the ZAMS model with a mass that is close to the mass of the model you want to calculate. Normally the ZAMS model will not have the starting parameters that you want, and as a consequence, you will have to rescale the ZAMS model.

NOTE: Because it is not possible to determine what opacities, surface boundary conditions, and equation of state were used to construct the ZAMS model, one should always relax the model for several non-evolutionary steps before starting the evolution.

With YREC you can rescale X, Z, the mixing length parameter, and the mass. You can rescale the core mass (for horizontal branch stars). You can rescale the Z in the core to a different value compared to the rest of the star. And you can reset the mass fraction location of the outermost shell in the interior (where it is fit to the envelope integrator).

If a parameter that controls rescaling, except CMIXLA and SENV0A, is set to zero or a negative number then no rescaling will be done.

RSCLM is used to rescale the mass. You can safely rescale a model in $0.1M_s$ increments. Because rescaling does not adjust convective regions, you cannot rescale, for example, from a ZAMS model that does not have a convective core to a slightly more massive model that does have a convective core. In this situation you should scale from higher masses, which have convective cores, down to the lower mass model that you want (which also has a convective core).

NOTE: YREC7 does not contain the modifications of Jamie Howard that enable the code to evolve massive stars ($10M_s$).

RSCLX and RSCLZ are used to rescale the X and Z mass fractions throughout the entire ZAMS star. If you rescale Z you must use the correct opacity tables for the final rescaled model. You can also rescale individual elements making up the Z in the first run only. This is an unsupported feature. See LNEWCP discussion.

XENV0A and ZENV0A are a relic of earlier versions of the program. They should be set to the final rescaled values of the initial surface abundance. They are used by YREC to set up the envelope opacity tables for 2D interpolations.

When rescaling composition, set the Z for the opacity and equation of state tables to match the final rescaled models.

You can reset the mixing length with CMIXLA. You should rescale the initial model's mixing length parameter before doing the evolution run. Many of the ZAMS models have quite different mixing length parameters. The mixing length parameter is determined from a tuned solar model. It depends on the opacities, composition, equation of state, and atmosphere model used in the calculation. Using OPAL and Kurucz opacities, Debye Hückel correction and a gray atmosphere requires CMIXLA=1.7. If you use a Krishna Swamy atmosphere then CMIXLA=1.9. Please ask one of the experts for the best value to use or consult the values from a tuned solar model listed in the Calibrated solar model section.

You can reposition the outer most mass shell of the Henyey integration (the interior) be setting LSENVOA to .TRUE. and then specifying the mass fraction *above* the outer most shell with SENVOA. You will run into numerical problems if you change the position, in one step, by too large an amount. The model will rescale OK by after a few evolutionary time-steps the model will fail to converge.

It is possible to specify a different Z in the core compared the rest of the star. This a nonstandard model feature. The parameters to control this option are ignored when LZRAMP=.FALSE. If you do want to rescale the Z in the core to a different value from the rest of the star then set LZRAMP=.TRUE. Specify the Z in the core using the RSCLZC parameter. The low Z core will extend to the mass fraction specified by the RSCLZM1 parameter. Then the Z will be linearly ramped to the outer Z value (RSCLZ) in the mass fraction range RSCLZM1 to RSCLZM2. From RSCLZM2 outward the outer Z value is used. When using this option, YREC will look for two sets of opacity tables so that in can carry out linear interpolation and extrapolation.

Rescaling a pre-main sequence model requires extra consideration. See next section.

Pre-main sequence rescaling

YREC can calculate pre-main-sequence models. The initial pre-main sequence models are generated using a program that solves the Lane-Emden equation. YREC can evolve such a model to the ZAMS without too much difficulty.

YREC cannot, though, rescale these models. YREC must rescale and evolve at the same time. That is to say, YREC must be allowed to evolve the model into its rescaled configuration. To rescale a pre-main sequence starting model set KINDRN=3. For this run you must specify both the rescaling parameters and the evolution parameters. All rescaling options described in the previous section can be applied to the model (of course you will not be able to lower Z in the core

because the pre-main sequence star is completely convective and will immediately mix the Z throughout the star.

The automatic time-stepping routines in YREC primarily use the rates of nuclear burning in the core to determine the size of the time-step. These routines do not work on the pre-main sequence because nuclear burning is not occurring. We recommend that you set LPTIME=.TRUE. in the $PHYSICS NAMELIST. This tells YREC to use an optional time-stepping routine. The parameters ATIME(8) through ATIME(10) in the $PHYSICS NAMELIST are used to tune the optional time-stepping routines.

Detailed Model Output or Pulsation Output Options

YREC7 contains several different options for producing a detailed output of the model's structure from the center all the way to the top of the atmosphere. This output was originally set up for use by Guenther's nonradial, nonadiabatic stellar pulsation program (jig), but, of course, it can be used for other purposes. Owing to this history, though, the parameters controlling this output are all associated with pulsation.

Three basic options exist for selecting which models during an evolutionary run are output: you can output the last model in the run; you can output models that are regularly spaced along the evolutionary track in the HR-diagram; and you can output models at specific ages during the evolution.

In addition to the detailed model, a corresponding *.short file is also output which summarizes the basic characteristics of the model. Separate programs exist which can read the pulsation model output and the *.short files and produce nicely formatted tables. When outputting detailed models along the evolution, individual files are used for each model. The files are automatically numbered in sequence, e.g., if FPMOD='solmod', then solmod_0000.pmod, solmod_0001.pmod, solmod_0002.pmod, etc. and solmod_0000.short, solmod_0001.short, solmod_0002.short, etc. will be output.

To output a detailed model for the last model of a run set LPULSE=.TRUE. and define the prefix filename FPMOD, and if IPVER=3, also FPENV, and FPATM. With IPVER=3, YREC will output a detailed listing of the model, the envelope, and the atmosphere to the files FPMOD, FPENV, and FPATM, respectively. A separate program is available to read these three files and produce a nicely formatted table of the model's structure. If IPVER=4, YREC will output the FPENV and FPATM data appended to the FPMOD file. That is, only one file will contain all the data. The data is not in an easily human-readable form. This format can be read by Guenther's pulsation program jig8. Earlier versions of the pulsation program (jig7, jig6, etc.) can only read IPVER≤3.

To output detailed model listings at equally spaced intervals along the evolutionary track in the HR diagram, set LPOUT=.TRUE. In addition to dumping out a detailed model, as described in the previous paragraph (and according to IPVER), YREC will also output a corresponding *.short file. The files are numbered sequentially starting with 0000. The 0000 file will always be the very first model of the evolutionary run, immediately following the rescaled initial model. You control the distance in the HR-diagram between models using the CONTROL NAMELIST parameters: POMAX, POA, POB, and POC. POMAX is the arclength (squared) of the track in the HR-diagram between outputted models. The arclength is defined by (POA * $\Delta\mathrm{Log}(L/L_s)$)^2 + (POB * $\Delta\mathrm{Log}(T_{\mathrm{eff}})$)^2 + (POC * $\Delta\mathrm{Age}$)^2, where $\Delta\mathrm{Log}(L/L_s)$ is the difference in log luminosity between the current model and the previously output model, and similarly for $\Delta\mathrm{Log}(T_{\mathrm{eff}})$ and $\Delta\mathrm{Age}$. When the arclength is greater than POMAX a model is output. Typical values for POMAX are listed in the example CONTROL NAMELIST. By adjusting POMAX one can control the density of models produced along the track. Note, though, that YREC will not output any more models then set by

the time-stepping routines. If you want to output models with a greater density than this, you will have to adjust the time-step parameters (ATIME in PHYSICS NAMELIST) to reduce the time-step size.

To output detailed models at specified times set LAGES=.TRUE. and define the ages (AGES(I) at which you want detailed models output. The total number of ages used must also be defined (NAGES). The first evolved model is automatically output so you do not need to define a 0.0 Gyr age. As YREC evolves it will automatically adjust its time-steping so that it hits the specified ages exactly (Note that YREC will continue to use its own optimum time-steps, but when it is within one time-step of a specified age, it will adjust the time-step so it hits the age). At each age a detailed model and short file will be output. Up to 100 different ages can be selected. Ultimately this feature could be used to generate models to build isochrones.

What to do if models fail to converge

NOTE: If YREC crashes or stops before it has even calculated or rescaled the first model then you have probably made an error in specifying the opacity files, the mass, the mixing length, the X, the Z, or the starting model. Check these things first. Look for typos in the NAMELIST.

NOTE: If YREC reports an unknown variable in the NAMELIST, then the error is a typo in that variable name in the NAMELIST. Most FORTRAN's will not tell you which variable contains the typo. If this is the case, we recommend moving the $END statement in the NAMELIST upward, testing the run, until YREC can read the NAMELIST successfully. The errant variable will then be located somewhere after the $END statement.

If you are using a modified version of YREC you are well advised to try to reproduce exactly the results of the unmodified code (with flags set to skip over your modifications) before you begin runs with your modified code. It is extremely easy to make a modification that affects a completely different section of the code, without your knowledge. When modifying the code, assume that you will probably make mistakes so put lots of debugging tests throughout.

The parameters required to evolve models of different mass through different phases of evolution are different. In other words, it is not possible, except in rare circumstances, to evolve a star all the way from the pre-main-sequence to the asymptotic giant branch with one set of parameters. NAMELISTs with parameters designed to work with various regions in the HR diagram have be developed by trial and error. You should seek out one of the experts and begin with one of these NAMELISTs as a starting point.

Prepare to spend anywhere from a day to a couple of weeks tracking down "failed to converge" problems. When a model fails to converge you will see in the FSHORT file some error message. The information that is output can be used by experts to diagnose the problem but in most cases it is misleading. For example, an error message indicating that you have stepped outside the opacity table may not be an error in the opacities but might well be an error in the time-stepping routines. Furthermore, because of the overlapping nature of the low and high temperature opacities, an error reported in the OPAL95 opacity table, for example, might be caused (actually very probably) by an error in the Alexander table.

If you do get an out of table error from one of the opacity routines, try running again with TOLALEX increased from 0.0D0 to 0.3D0 (or more). This allows the code to extrapolate beyond the edge of the low temperature Alexander table in log T and density (normalized values) by up to 0.3 in the log (a factor of two). I have successfully extrapolated up to a factor of ten (TOLALEX=1.0D0) in my own tests. You can also set a similar extrapolation factor for the OPAL95 tables (TOLOPAL95). Only the Alexander and OPAL95 tables have this feature

(although, it would be straightforward to modify the other opacity routines to allow extrapolation).

A common opacity fault occurs when evolving near or on the giant branch. Even though no appreciable Z burning is taking place, trace amounts of Z can burn. The code will notice that Z in the core is different from Z at the surface and will look for a pure Z table so it can determine the opacity by interpolation in Z. The code will stop at this point. To carry out the evolution without halting set LPUREZ=.TRUE. and specify the location of a pure Z opacity table.

If your models fail to converge at some point, then redo the run stopping about ten models before the point at which the model diverges. Use the *.last model as your starting model and begin the trial and error testing from this point. Turn on the FMODPT output (normally kept off because it generates huge disk files) and examine the output to see if you can discern the cause of the divergence.

Try reducing the time-step size. Before you can do this you need to examine the FSHORT file to determine the phase of evolution the star is in. Then you can adjust the time-stepping parameter (ATIME) appropriate for that phase of evolution (e.g., it does no good to adjust the helium shell burning criterion when you are on the main sequence.)

You can also try increasing the number of shells in the model by adjusting the HPTTOL parameters. Again you must use common sense to adjust the parameters relevant to the phase of evolution you are in. Note that the maximum number of shells in the interior of the model is set to 2000. This can be changed but requires re-compiling YREC.

NOTE: you must set ITER3=ITER4=0 to evolve stars up the giant branch. If you do not, you will notice the track up the giant branch is quite jagged.

As mentioned in the previous section, YREC's default time-stepping routines rely on nuclear reaction rates to control the time-stepping. When the central regions of the star are contracting (e.g., at turnoff) then the routines may not slow the evolution down enough to follow the rapid changes in structure. You need to adjust the ATIME parameters to slow the evolution down during this phase of evolution. You can also try the optional time-stepping routines by setting LPTIME=.TRUE.

If the models diverge on the very first model after rescaling or shortly thereafter (and you are not on the pre-main-sequence) then you may have specified an incorrect opacity table (that does not match the Z of the model) for one of the tables. Or you may have rescaled the initial model by too large a factor. Try rescaling to the desired model in smaller steps.

It takes the experts anywhere from a couple of hours to a couple of weeks to solve divergence problems. Some divergence problems have not been resolved (e.g. the ITER3-ITER4 problem on the GB). In extremely rare cases the divergence problem may be consequence of a bug in YREC in which case only those familiar with the workings of the YREC (Demarque, Pinsonneault, and Guenther) program are going to be able to solve the problem.

Running YREC on UNIX machines

The NMLCONTROL NAMELIST file must be in the same directory as your copy of YREC (the executable) and it must have the name "yrec7.batch".

On unix machines to run the program using the background feature of the cshell, type:

```
nice yrec7 >! yrec7.log &
```

The "nice" ensures that the program does not hog the CPU. the ">! yrec7.log" will create a file with the name "yrec7.log" that will contain a running log of the program's execution (the "!" will erase any existing file of that name before creating a new one). The "&" at the end tells the C shell that you are submitting the job so that it will run in the background. If you are not familiar with these commands they are all described in the *man* entry on cshell.

In UNIX you can freely inspect any file that is being used by YREC while YREC is running. For example, to see your progress you can type:

```
more example.track
```

to examine the track file. Note that on most computer systems the output is buffered before outputting to the disk, hence, the file will be written to in spurts. If the program stops, diagnostic information will be written by YREC to the *.short" file and to the "YREC.log" file.

Execution speed

The rate at which YREC executes depends on a number of factors. When submitting long runs it is desirable to estimate the length of the run. Here we provides some facts which will enable you to estimate execution time.

The time to calculate each time-step (no diffusion or rotation), is proportional to the square of the number of shells used in the model.

On a DEC Alpha it takes less than 1s to calculate a model with 600 shells. If you are using the OPAL equation of state then the time is increased by a factor of 2 to 3.

If you start on the main sequence with 200 to 300 shells in the model, the number of shells will grow (as the hydrogen burning shell thins) on the giant branch to 900–1200 shells.

It will take ~50 models to evolve from the ZAMS to turnoff, then it will take another ~300 models to evolve up the giant branch to near the luminosity of the horizontal branch, and then it will take ~2000 models to reach the tip of the giant branch and the ignition of helium burning. These numbers are approximate and can vary by more than a factor of two.

YREC requires approximately 10MBytes of memory to run with JSON=2000. (JSON is the maximum number of shells allowed in the model).

If you use the OPAL eos tables, for improved accuracy of the physics of the eos, then execution speed is reduced by a factor of approximately 3!

Modifying YREC

Please obtain permission from Demarque, Pinsonneault, or Guenther before making any modifications. They will direct you to the current stable version of YREC.

Please identify all your modifications with comments and rename the YREC program so that it will not be confused with the original version of the program, such as YRECa. Please do not use the name YREC8, YREC9, etc., as they are reserved for the next versions of YREC. All modifications should be controlled by a logical flag, e.g. `LBUZZ`, if `LBUZZ` is `.TRUE.` then execute your modified section of code, if `LBUZZ` is `.FALSE.` then skip your modifications. After you have

made your modifications test that the code with the logical flag set to skip your modifications, i.e. make sure that if `LBUZZ=.FALSE.` the results are exactly the same as the original code. And by exactly, we mean that the numbers should agree to the last decimal place. If they don't you have introduced an error into the code.

We recommend that you place all the new variables you need to pass in and out of existing subroutines in a `COMMON` block.

And most importantly, when modifying YREC, assume that your modifications will be used by others for the next several decades.

Here is a Mr. Politeness Programmer check list:

- Identify *all* sections of code you write with a comment card such as:

C JULES VERNE 3/95 CARBON BURNING ROUTINES

  which identifies your name, the date of the modification, and the nature of the modification
- Do not delete obsolete code, comment it out only.
- Do not use mixed case variable names. The code has a history dating back to the 1960's hence was written with capital letters. Please stick to this convention and set your compiler to ignore case. That is, for example, lpulse should be the same variable as LPULSE or Lpulse, etc.
- Use WRITE statements (no PRINT or TYPE statements)
- Do not modify the argument list of any subroutine. Pass your variables using common blocks.
- Use lots of comments.
- Before introducing a new variable do a global search through the source code to make sure it is not already being used.
- Use double precision variables only. The only sections that are in single precision are some of the older opacity routines.
- Variables beginning with I, J, K, M, and N are assumed to be of type INTEGER, and variables beginning with L are type LOGICAL. Don't override this convection.

The best way to learn how to modify the code is to examine some recent modifications. To see how extensive output is done examine the pulsation output routines (LPULSE) which pull variables from the interior, envelope and atmosphere. If you want to change the opacities examine the implementation of the Kurucz opacities (LKUMOL), or equation of state, check out the implementation of OPAL eos (LOPALE) or Debye-Hückel (LDH).

Unfortunately, text editors vary in how they handle tabs. As a consequence, much of the well-intentioned indenting used to help identify structured programming segments in YREC are now, after many revisions by many programmers using many different text editors, poorly indented. We recommend, if possible, to turn-off tabbing in your editor and use spaces instead for indenting.

Program Constants

The following physical constants (in cgs units) are defined in the subroutine SETUPS and are used throughout YREC.

Luminosity of the Sun:	CLSUN= 3.8515D33.
Mass of the Sun:	CMSUN = 1.9891D33.
Radius of the Sun:	CRSUN = 6.9598D10.
Bolometric magnitude of the Sun:	CMBOL = 4.79D0 (N>B> depends on b.c. zero-point !)
Number of seconds per year:	CSECYR = 3.1558D7.
Speed of light:	CC = 2.99792458D10.
Stefan-Boltzmann constant:	CSIG = 5.67051D-5.
Molar gas constant:	CGAS = 8.3145l0D7.
Gravitational constant:	G = 6.67259D-8.
Mass of the electron:	CMELEC = 9.1093897D-28.
Bolztmann constant:	CBOLTZ = 1.380658D-16.
Planck's constant:	PLANCK = 6.6260755D-27.

Opacities and Equation of State

Introduction

Because the current opacity situation is complicated we devote an entire section to describing how the opacities are used.

We recommend that you use the OPAL95 opacities (LOPAL95=.TRUE.) with the latest interpolation routines (LNEWOPAL95=.TRUE.) along with the Ferguson (0 LFERG05 = .TRUE.) low temperature opacities. These are the most recent tables. These tables will automatically interpolate to the model Z. Z does not have to be specified for the tables since the entire opacity table is read in and the Z portion is computed. Other opacity tables are provided for backward compatibility only.

The low temperature opacities are used for log T < 4.0. In the transition region 4.0 < log T < 4.1 the low temperature opacities tables and the interior opacities are ramp averaged. Above log T = 4.1 the interior opacities are used. Both the Alexander 95 opacities and the OPAL 95 opacities will automatically interpolate to the required Z (e.g. ZALEX1). For the other opacities, the appropriate opacity file table nearest the required Z must be explicitly read in.

If you are calculating models with Z diffusion LDIFZ=T or are calculating non-standard solar models with variable Z in the core LZRAMP=T then a second set of opacity tables must be read in if you are using older tables. YREC will interpolate between the two different tables to obtain the opacity at the required Z for each shell. Internally, the logical flag L2Z is used. L2Z= LZRAMP.OR.LDIFZ and is defined in subroutine "setupopac".

During HB evolution Z in the core will increase to 1.0. The opacity in this region is calculated by interpolating between the interior Z opacity table and a pure Z opacity table. You must explicitly set LPUREZ=T to read in the pure Z table. The table is an LAOL 89 opacity table but can be used with any of the other interior opacity tables.

If the density or temperature is outside the range of the opacity tables, or outside the allowable extrapolation range, YREC will stop and report an error message in FSHORT. If during an evolutionary run you get a warning message stating that you have stepped outside the opacity table, do not assume you need more extensive opacity tables. This message is warning you that YREC was trying to look up an opacity for a temperature and density that were outside the tables (assuming that TOLALEX=0.0D0 and TOLOPAL95=0.0D0). In most cases, the fundamental cause of the warning has to do with a diverging model where, during the iterative procedure, the model has started to "blow up." The diverging model may have ridiculously high or low

temperatures and densities. The warning message will output these numbers. If they are for example, unrealistically high, logT > 10.0, then you can assume it is not a problem with the opacities. You need to fix the problem at the source by reducing time-steps or increasing the number of shells in the model to facilitate convergence.

THIS PARAGRAPH IS OBSOLETE: The recommended tables do not require this. (There may be cases when you really do step out of the tables. If you are using the OPAL95 or Alexander opacities you can stretch the boundaries of the tables by allowing YREC to extrapolate beyond the edge of the tables. To do this you specify the amount, in the log, using the TOLALEX and TOLOPAL95 parameters in the CONTROL NAMELIST. For example, if you set TOLALEX=0.3D0 then YREC will extrapolate in log normalized density and log normalized temperature, 0.3 (a factor of 2) beyond the edge of the table. The extrapolation is linear. Because the low temperature tables are accessed before the high temperature tables, you should first increase TOLALEX in increments of 0.3 when trying to get YREC to produce opacities near the edge of the tables. I have tested values of TOLALEX up to 1.0 (a factor of 10). If increasing TOLALEX does not work, then try increasing TOLOPAL95. Make every effort to determine exactly which tables and how for beyond the edge you are going. Also, remember to return TOLALEX and TOLAOPAL95 to 0.0 after you have evolved through the "rough" patch.

OPACITY COMMON BLOCKS

```
      COMMON /NEWOPAC/ZLAOL1,ZLAOL2,ZOPAL1,ZOPAL2, ZOPAL951,
     +       ZOPAL952, ZALEX1, ZALEX2, ZKUR1, ZKUR2,
     +       LLAOL89,LOPAL92,LOPAL95,LKUR90,LALEX95,L2Z
      COMMON /MISCOPAC/IOKUR2, FKUR2
```

OPAL95 OPACITIES

```
      COMMON /GLLOT95/GRIDT(NUMT),GRIDX(NUMX),GRHOT3(NUMD)
      COMMON /LLOT95/OPACITY(NUMXT,NUMD),NUMXM,NUMTM
      COMMON /GLLOT952/GRIDT2(NUMT),GRIDX2(NUMX),GRHOT32(NUMD)
      COMMON /LLOT952/OPACITY2(NUMXT,NUMD),NUMXM2,NUMTM2
```

splines

```
      COMMON /INTPL95/ CFORD(NUMXT,NUM4D),NDS(NUMXT),NDD(NUMXT)
      COMMON /INTPL952/ CFORD2(NUMXT,NUM4D),NDS2(NUMXT),NDD2(NUMXT)
```

surface tables

```
      COMMON /LLOT495/XENV4,ZENV4,CFORD4(NUMT,NUM4D),KIPM1
      COMMON /LLOT4952/XENV42,ZENV42,CFORD42(NUMT,NUM4D),KIPM12
```

OPAL92 OPACITIES

```
      COMMON /GLLOT/GRIDT(NUMT),GRIDX(NUMX),GRHOT3(NUMD)
      COMMON /LLOT/OPACITY(NUMXT,NUMD),NUMXM,NUMTM
      COMMON /GLLOT2/GRIDT2(NUMT),GRIDX2(NUMX),GRHOT32(NUMD)
      COMMON /LLOT2/OPACITY2(NUMXT,NUMD),NUMXM2,NUMTM2
```

splines

```
      COMMON /LINTPL/ CFORD(NUMXT,NUM4D),NDS(NUMXT),NDD(NUMXT)
      COMMON /LINTPL2/ CFORD2(NUMXT,NUM4D),NDS2(NUMXT),NDD2(NUMXT)
```

LAOL89 OPACITIES

```
      COMMON/NWLAOL/OLAOL(12,104,52), OXA(12), OT(52), ORHO(104),
     *  TOLLAOL, IOLAOL, NUMOFXYZ, NUMRHO, NUMT, LLAOL, LPUREZ,
     *  IOPUREZ, FLAOL, FPUREZ
```

```
      COMMON/NWLAOL2/OLAOL2(12,104,52), OXA2(12), OT2(52),
     *               ORHO2(104), NXYZ2,NRHO2,NT2
```

splines:

```
      COMMON/SLAOL/SLAOLL(12,104,52),SRHOL(12,104,52),
     *  SDORL(12,104,52), NUMRS(12,52)
      COMMON/SLAOL2/SLAOLL2(12,104,52),SRHOL2(12,104,52),
     *  SDORL2(12,104,52), NUMRS2(12,52)
```

ALEX OPACITIES

```
      COMMON /GALOT/TGR(NUMT),XXG(NUMX),R0GR(NUMD)
      COMMON /ALOT/CAPPA(NUMXT,NUMD),NUMXM,NUMTM
      COMMON /GALOT2/TGR2(NUMT),XXG2(NUMX),R0GR2(NUMD)
      COMMON /ALOT2/CAPPA2(NUMXT,NUMD),NUMXM2,NUMTM2
```

splines

```
      COMMON /INTPAL/ CFORD(NUMXT,NUM4D),NDS(NUMXT),NDD(NUMXT)
      COMMON /INTPAL2/ CFORD2(NUMXT,NUM4D),NDS2(NUMXT),NDD2(NUMXT)
```

KURUCZ OPACITIES

```
      COMMON /GKRZ/GRIDT(NUMT)
      COMMON /KRZ/OPACITY(NUMXT,NUMD),RHO(NUMXT,NUMD),NUMTM
      COMMON /GKRZ/GRIDT2(NUMT)
      COMMON /KRZ/OPACITY2(NUMXT,NUMD),RHO2(NUMXT,NUMD),NUMTM2
```

splines:

```
      COMMON /INTPL/ CFORD(NUMXT,NUM4D),NDS(NUMXT),NDD(NUMXT)
      COMMON /INTPL2/ CFORD2(NUMXT,NUM4D),NDS2(NUMXT),NDD2(NUMXT)
```

Opacity Subroutines

The latest Ferguson low temperature opacities and the improved interpolation OPAL95 opacities are read in in their entirety. Interpolation in X and Z is done during the evolution. That is, you do not need to specify two distinct Z's when evolving with Z diffusion. The comments below on this subject (in italics) refer to older opacities.

Three subroutines have been created to handle the opacities.

SETUPOPAC(XENV, V). This routine calls the appropriate subroutines to read in the requested tables. Routines to calculate the splines of the tables are also called. For OPAL 95, OPAL 92 and Alexander tables, subroutines to calculate a special set of surface opacity tables (at the current value of XENV) are called. *If LDIFZ=T or LZRAMP=T then a second set of opacity tables are also set up. L2Z is defined here as L2Z = LZRAMP.OR.LDIFZ.*

SURFOPAC(XENV). This routine assumes that the opacity tables have already been read in, i.e., SETUPOPAC has already been called. This routine re-calculates surface opacity tables (at the current value of XENV). This routine is called by HPOINT when helium diffusion changes the value of X in the outer convective envelope. The surface opacity table X is stored in XNEW which is passed by common block.

GETOPAC(DL,TL,X,Z,O,OL,QOD,QOT) This routine calculates the opacity O, the log of the opacity OL, the logarithmic derivative of O w.r.t. density, QOD, and the logarithmic derivative of O w.r.t. temperature, QOT, given log(T) and log(rho), X, and Z.

1. The routine first determines the low temperature opacity. *If L2Z=T then the opacity from the second opacity table is calculated and the two opacities are linearly interpolated to obtain the opacity at the requested Z.*
2. If T>10000K then will get an interior opacity.
3. First check to see if need to use the pure Z opacity table. The pure Z table is used if Z differs from the original ZENV, as will happen during HB evolution. If need to use the pure Z table, get opacity of pure Z then get opacity from specified interior opacity. Finally, interpolate to get opacity at Z. Otherwise, get opacity from specified interior opacity table. *If L2Z=T then the opacity from the second opacity table is calculated and the two opacities are linearly interpolated to obtain the opacity at the requested Z.*
4. If in the temperature range 4.0<logT<4.1 then ramp between the opacity obtained from the low T tables and the opacity obtained from the interior tables.
5. Calculate conductive opacity correction.

OPAL95 New interpolation and Ferguson OPACITIES

These opacity routines include a 4D interpolation (density, temperature, X, and Z). The code will read in the entire opacity table and carry out interpolations as required. For efficiency interpolated tables are stored at fixed X and X (when no diffusion exists in the envelope) so that only 2D interpolation is needed. (The slow bit is looking up where you need to do the interpolation into the tables, not the actual interpolation itself).

OPAL95 OPACITIES (in file opal95.f)

LL95TBL reads in table (second table if L2Z=T)

YLLOC95 sets up splines (second splines if L2Z=T)

LL4TH95 sets up surface table (second table if L2Z=T)

YLL2D95 2D interpolation of splines

YLL2D952 2D interpolation of splines second table

YLL3D95 3D returns opacity

YLL2D95 2D interpolation of splines

YLL3D952 3D returns opacity second table

YLL2D952 2D interpolation of splines second table

OPAL92 OPACITIES (in file opal92.f)

SETLLO reads in table (second table if L2Z=T)

YLLOC sets up splines (second splines if L2Z=T)

LL4TH sets up surface table (second table if L2Z=T)

YLL2D 2D interpolation of splines

YLL2D2 2D interpolation of splines second table

YLL3D 3D returns opacity

YLL2D 2D interpolation of splines

YLL3D2 3D returns opacity second table

YLL2D2 2D interpolation of splines second table

LAOL89 OPACITIES (in file laol89.f)

RDLAOL reads in table (second table if L2Z=T)

SULAOL sets up splines (second splines if L2Z=T)

RDZLAOL reads in pure Z table

ZSULAOL sets up splines for pure Z table

GTLAOL returns opacity

GTLAOL2 returns opacity for second Z table

GTPURZ returns opacity from pure Z table

ALEX OPACITIES (in file alex.f)

ALXTBL reads in table (second table if L2Z=T)

YALOC sets up splines (second splines if L2Z=T)

ALX8TH sets up surface table (second table if L2Z=T)

YALO2D 2D interpolation of splines

YALO2D2 2D interpolation of splines second table

YALO3D 3D returns opacity

YALO2D 2D interpolation of splines

YALO3D2 3D returns opacity for second Z table

YALO2D2 2D interpolation of splines second table

KURUCZ OPACITIES (in file kurucz.f)

SETKRZ reads in table (second table if L2Z=T)

YKOEFF sets up splines (second splines if L2Z=T)

KURUCZ returns opacity

KURUCZ2 returns opacity for second Z table

Additional Notes

Debye-Hückel correction to equation of state

The Debye-Hückel correction to the equation of state (Clayton 1976) attempts to take into account the Coulomb interactions of the plasma. The specific form implemented in YREC is applicable in the regime of high temperature and low densities and assumes that the gas is fully ionized. This, therefore, is not the general form of the D-H correction used in detailed equation of state calculations. If LDH = .TRUE. then for temperatures above log T = 6.0 YREC will apply a D.H. correction (to the density) when calculating the equation of state. Because many thermodynamic variables are calculated using analytic expressions, these are also modified to be self consistent. There are other Coulombic corrections that are not included that may be equally important so use the D.H. correction with caution.

Solar *p*-mode oscillation comparisons show that solar models that use the OPAL equation of state tables are more accurate, according to high-*l* *p*-modes, than models that use the analytic form with the D.H. correction, which in turn are more accurate than models that just use the analytic form.

Krishna Swamy Atmosphere model

The Krishna Swamy T-τ relation is a theoretically based empirical fit to the Sun's observed T-τ dependence in the lower atmosphere. The relation follows the drop in T from the surface of the photosphere, where $T_{\rm eff}$ is defined, at τ=0.312156330, to τ=10^{-4}. It does not model the temperature minimum at the base of the corona nor the temperature rise above. It is given by:

$$T^4 = 0.75\ T_{\rm eff}^4\ (\tau + 1.39 - 0.815\ e^{-2.54t} - 0.025\ e^{-30t})$$

Conductive opacities

Description needed.

Convective Overshoot and Semi-convection

For convective overshoot the convective boundary is stable against local mixing, and some mixing across the formal boundary does not change the boundary location. The buoyancy force vanishes at the boundary. However, the convective velocities do not vanish and therefore some penetration can take place into the radiative layers.

The code then extends mixing of the chemical composition beyond the formal boundary of the convective zone, as defined by the Schwarzschild criterion, by a chosen fraction of the local pressure scale height at the convective boundary. In this fully mixed overshoot region, the local temperature gradient is set by the Schwarzschild criterion, and is therefore radiative.

If mixing across the formal radiative-convective boundary causes convective instability on the radiative side, the region is said to be unstable against semi-convection. The code then allows the convective region to grow into the radiative layers to a point where the convective boundary is stabilized against composition perturbations.

Nuclear reaction rates

Some of the nuclear energy generation routines and cross sections are from Bahcall & Pinsonneault (Bahcall, J. N. & Pinsonneault, M. H. 1992, RevModPhys, 64, 885; Bahcall, J. N. & Pinsonneault, M. H. 1992, ApJ, 395, L119; Bahcall, J. N. 1989, Neutrino Astrophysics, Cambridge Univ: Cambridge) . The new routines are only called in the third and fourth iteration levels NITER3, NITER4. The new cross sections are used throughout. NITER3 and NITER4 cause numerical instabilities for evolution up the giant branch and should not be used.

An error was discovered in the rates calculation (engeb_mp.f). It is noted by the comment line C 2009 Dec Fixed DBG.

Diffusion

Diffusion of elements appears to be an important process that occurs in some stars owing primarily to the gravitational settling of heavier than hydrogen elements—the heavier than hydrogen atoms sink with respect to the lighter hydrogen atom. YREC can model helium diffusion and Z diffusion. Several formulations have been programmed. See Brian Chaboyer if you want to test other formulations or if you want to consider the combined effects of rotation and diffusion (rotation inhibits diffusion).

The diffusion calculation is an extra iterative calculation between each time-step. The code will automatically calculate the number of extra diffusion time-steps necessary.

Recommended settings (in PHYSICS NAMELIST, a.k.a. *.nml2 files)

Helium Diffusion:

```
LDIFY=.TRUE.
LDIFZ=.FALSE.
LTHOUL=.FALSE.
LTHOULFIT=.FALSE
ILAMBDA=3
FGRY=1.0D0
FGRZ=1.0D0
DT_GS = 0.1D0
XMIN = 1.0D-3
YMIN = 1.0D-3
GRTOL = 2.0D-7
NITER_GS = 10
```

Z Diffusion:

```
LDIFY=.TRUE.
LDIFZ=.TRUE.
LTHOUL=.TRUE.
LTHOULFIT=.FALSE
ILAMBDA=4
FGRY=1.0D0
FGRZ=1.0D0
DT_GS = 0.1D0
XMIN = 1.0D-3
YMIN = 1.0D-3
GRTOL = 2.0D-7
NITER_GS = 10
```

Numerical Accuracy

It is believed, that YREC can produce numerically models that are accurate to about 9 to 10 significant digits without major modification.

Wisdom

The teeth are hard and fall out; the tongue is soft and remains.

Rotation

Introduction

M. H. Pinsonneault modified the existing Yale stellar evolution code so that it could model the effects of slow rotation. These codes are included in YREC. They should only be used after reading Pinsonneault's thesis.

Detailed description of NAMELIST files

NOTATION NOTE: Default values are listed in "{}". Values for pulsation, if different from the standard values, are listed in "[]".

Warning: there are probably errors in this documentation. Use common sense.

$NMLCONTROL NAMELIST ("yrec7.batch")

YREC7 expects to find the file "yrec7.batch" located in the same directory as the executable program itself. This is the first file YREC7 reads.

NUMNML
: The number of CONTROL and PHYSICS NAMELIST pairs defined. YREC will read each pair of NAMELISTS and carry out the runs defined in those NAMELISTS, one after the other.

FNML1(I), FNML2(I), I=1, 100
: The full file names for the NAMELIST pairs. FNML1 identifies the CONTROL NAMELIST and FNMML2 identifies the PHYSICS NAMELIST.

$CONTROL NAMELIST (*.nml1)

DESCRIP(I), I=1,2
: Declared as CHAR*256. Can be used to document your run with a description.

Solar and stellar calibration parameters

LCALS {.FALSE.}
: If .TRUE. then YREC will calculate a calibrated solar model. The first run, a rescaling run, and the second run, an evolution run, must be set up by the user. YREC will generate additional runs while it attempts to find the right helium abundance and mixing length parameter to produce a solar model.

TOLR

If LCALS=T then a solar model will be calculated with Log $R/R_S \leq$ TOLR.

TOLL

If LCALS=T then a solar model will be calculated with Log $L/L_S \leq$ TOLL.

LINCLUDEZX {.FALSE.}
ZOVERX
TOLZX

If LCALS=T and LINCLUDEZX=T then also try to tune Z/X to ZOVERZ within tolerance TOLZX

LCALST {.FALSE.}

If .TRUE. then YREC will calculate a calibrated stellar model of specified mass, radius (or effective temperature), and luminosity. The first run, a rescaling run, and the second run, an evolution run, must be set up by the user. YREC will generate additional runs while it attempts to find the right helium abundance or mixing length parameter to produce a stellar model.

XLS, XLSTOL

If LCALST=T then a stellar model will be calculated with L/L_S = XLS±XLSTOL

LTEFF

If LCALST=T then if LTEFF=T use STEFF to specify the desired effective temperature of the model or if LTEFF=F use SR to specify the desired radius of the model.

STEFF

If LCALST=T and LTEFF=T then a stellar model will be calculated with T_{eff} = STEFF.

SR

If LCALST=T and LTEFF=F then a stellar model will be calculated with R/R_S = SR.

LADJX

If LADJX=T then will adjust the helium abundance (or equivalently the hydrogen abundance) to produce a tuned model. If LADJX=F then will adjust the mixing length parameter.

Rescaling and evolving kind card parameters

NUMRUN

Specifies the number of runs to be carried out in the kind card loop. Rescaling is counted as one run. The arrays within YREC are set so that up to 50 runs can be carried out in one kind card loop.

LZRAMP {.FALSE.}

If .TRUE. then rescale the starting model to have a different Z in the core (RSCLZC). The Z in the initial model is set to RSCLZC from the center to mass fraction RSCLZM1. Then Z is linearly ramped up to the envelope Z, RSCLZ, from RSCLZM1 to RSCLZM2. And then from RSCLZM2 to the surface the initial model is rescaled to the envelope Z, RSCLZ. This option requires a second set of opacity tables so that opacities can be calculated for all Z by interpolating the two tables.

KINDRN(I), I=1,...,50

Specifies whether the Ith run is an evolutionary run, viz KINDRN(I)=1, a rescaling run, viz KINDRN(I)=2, or a rescaling plus evolutionary run, viz KINDRN(I)=3. The array index corresponds to the number of the run, i.e. first run, second run, etc. The rescaling plus evolutionary run (3) is required to rescale a pre-main sequence model. Other run types can be easily added.

LFIRST(I), I=1,...,50

Specifies whether the Ith run will use the previous run's last model as the starting model, viz LFIRST(I)=.FALSE., or the Ith run will read the input model (IFIRST) as the starting model, viz LFIRST(I)=.TRUE..

For example, one begins by reading a model from a disk file so that LFIRST(1)= .TRUE.. If the first run is a rescaling run, i.e. KINDRN(1)=2, then the read-in model is rescaled. To use the rescaled model in the next run set LFIRST(2)=.FALSE. This indicates that the last model calculated in run 1 will be used as the starting model of run 2. If in the third run you want to evolve directly from the model originally read from a disk file then specify LFIRST(3)=.TRUE. YREC will then reread the model from the disk and use it as the starting model for the third run.

RSCLM(I), RSLCX(I), RSCLZ(I), RSCLCM(I), I=1,...,50

Specifies the rescaled values of mass, RSCLM(I), hydrogen abundance, RSCLX(I), metal abundance, RSCLZ(I), and the core mass, RSCLCM(I) of the Ith run when KINDRN(I)=1, i.e. the run is a rescaling run. The values are ignored if KINDRN(I)=2, or if the rescaled value is less than or equal to zero.

For example, if KINDRN(3)=1 and we have RSLCX(3)=0.0, RSCLZ(3)=0.0200, RSCLM(3)=-1.00, and RSCLCM=-0.30 then YREC on the third run will rescale the input model's Z to Z=0.0200. The other values will not be rescaled.

More than one parameter can be rescaled at a time. Experience shows that one should rescale in small steps. Rescaling a 1.00 M_S model to 0.5 M_S in one step would probably fail.

Rescaling occurs in two steps: first the old value is replaced by the new value in the model, then the model is "relaxed", that is the model is run through the stellar evolution equations with the entropy term set to zero (evolution is turned off). The result is a physically self consistent stellar model. (NOTE: the model may not make sense in terms of stellar evolution, in that there may be no physical way to arrive at the rescaled configuration.)

Experience shows that all ZAMS models should be relaxed before beginning an evolution run. This is automatically performed if the model number of the model is negative. We recommend that the ZAMS model number be set to -1.

RSCLZC(I), RSCLZM1(I), RSCLZM2(I)

If LZRAMP=T and KINDRN=2 then will rescale the Z form the center to mass fraction RSCLZM1 to RSCLZC. The Z from the mass fraction RSCLZM2 to the surface will have the Z specified by RSCLZ. Z is ramped linearly between the two values of Z from RSCLZM1 to RSCLZM2.

NMODLS(I), I=1,...,50

Specifies the maximum number of models calculated in the Ith run. This number applies to both evolutionary runs and rescaling runs. In a rescaling run the model will be rescaled once then relaxed, i.e. run through the stellar evolution equations with the entropy term set to zero (no evolution), NMODLS(I) times.

Parameters which must be specified for every run

XENV0A(I), ZENV0A(I), I=1,…,50

XENV0A(I) specifies the initial hydrogen abundance *X* at the surface and ZENV0A(I) specifies the initial metal abundance *Z* at the surface of the ZAMS model for the Ith run. These mass fraction abundances represent the primordial abundance of the star being evolved and are used in YREC to set up the 2D opacity tables for the envelope.

CMIXLA(I), I=1,…,50

Specifies the mixing length parameter for the Ith run. This characterizes the efficiency of convective energy transport. Experience shows that values between near 2 are reasonable for YREC (other codes using different mixing length formalisms will likely require different mixing length parameters). For detailed models of the Sun containing about 1500 shells divided evenly among the interior, envelope, and atmosphere and using Z=0.0188 (Anders and Grevesse mixture 1989) and OPAL and Kurucz opacities with Debye Hückel, and Krishna Swamy atmosphere requires CMIXL=1.9. For a gray atmosphere the solar model requires CIMXL=1.7 (see Calibrated solar models section for more values).

You should not do a stellar evolutionary calculation until you know what to set CMIXL to. If you don't know then ask. The mixing length parameter controls the radius of stars with convective envelopes. An incorrect mixing length parameter will give meaningless radii or equivalently meaningless surface temperatures.

LSENV0A(I), I=1,…,50; SENV0A(I), I=1,…,50 {50*F, 50*2.0D-4}

The stellar evolution program divides the model into the interior and the envelope. The envelope starts at mass fraction SENV0A. SENV0A, called the fitting point where the interior integration is matched to the envelope integration, does not change during an evolutionary run. If you want to change it, i.e., move the fitting point inwards or outwards, then set LSENV0A(I) to .TRUE. for the run and specify the mass radius of the fitting point for the Ith run by setting SENV0A(I).

You cannot make a large change to the fitting point in one go. Also, values smaller than 1.0D-12 make the code unstable.

Basic output flags

In the current version of YREC all diagnostic information is output to FSHORT and all model information (i.e. summaries of structure) is output to FMODPT. This differs from all previous versions of the stellar evolution code in which it was anyone's guess as to what was output to FSHORT and what was output to FMODPT. YREC7 expands on the optional pulsation output by providing the option to produce the detailed output, suitable for pulsation studies, at specific ages. Of course, the detailed models do not have to be just used for pulsation. The detailed model output lists structure variables extending from the core, through the interior, envelope, out to the top of the atmosphere. A separate program is required to read the detailed model output and cast it in a form suitable for human consumption.

LTRACK

If .TRUE. then a summary of the model is output to the FTRACK file, otherwise it is not. After each model is calculated the following information will be saved to the FTRACK file, for version 1, i.e. ITRVER=1: model number; number of shells in model; age; luminosity; radius; gravity; effective temperature; convective core mass; convective envelope mass; central temperature, density, pressure, ratio of gas pressure to total pressure, electron gas degeneracy, hydrogen mass fraction, helium mass fraction, and metals mass fraction; luminosity of ppI, ppII, ppIII, CNO, triple-alpha, He to C, gravity, and neutrinos; central and surface abundances of ^{3}He, ^{12}C, ^{13}C, ^{14}N, ^{15}N, ^{16}O, ^{17}O, ^{18}O and surface abundances of (if LEXCOM = .TRUE.) ^{2}H, ^{6}Li, ^{7}Li, ^{9}Be; the individual neutrino luminosities (if LNEWE =.TRUE.); and the mass fraction and radius fraction locations of the H burning shell's bottom, midpoint, and top.

If ITRVER=2, then additionally, the following rotation information is output: total angular momentum, total kinetic energy of rotation, surface angular velocity, and the central angular velocity.

If ITRVER=3, then only one line of information is output. This form is useful for plotting and saving space.

ITRVER

To facilitate future extensions to the FTRACK output file without jeopardizing compatibility with existing versions each FTRACK file is identified by a version number. This number is saved in the header of the FTRACK file.

If ITRVER=1 then the default information is output (see LTRACK above). If ITRVER=2 then in addition to the default output rotation information is included (see LTRACK above). ITRVER = 3 then only one line of information is output per model, which is useful for reading into a plotting program.

LSTORE, NPUNCH

If LSTORE is .TRUE. then output a stellar model to the file FSTOR, otherwise do not. This is an ASCII text file which has the form of an input model file. Every NPUNCHth models will be saved. If NPUNCH ≤ 0 then no models will be saved. The models are appended to the end of the FSTOR file. This form of output is totally useless; better to use the binary output option if you want to save every n'th model. The word "PUNCH" in this option should give you an idea how old this option is—originally, the models were read in on punched cards, and this option output models on punched cards.

LSTPCH

If .TRUE. then output the last stellar model calculated on each run to the file FLAST, otherwise do not. This is an ASCII text file which has the form of an input model file. Note that the way YREC saves the last model is simply to save every model calculated while `REWIND`ing the FLAST file just before saving. (LSTPCH comes from "Last Punch")

LSCRIB

If .TRUE. then output model information to file FMODPT according to how the NPRT1, NPRT2, NPRTPT, LPSHLL, LCONZ0, LCHEM0, LJOUT, LPRTIN, NPENV, LCORR, NPOINT parameters are set. If .FALSE. turn off all output to this file.

NPRT1

Output basic model information every NPRT1th model to FMODPT. If NPRT1 < 0 then do not output anything.

NPRT2

Output additional model information every NPRT2th model to FMODPT If NPRT2 < 0 then do not output anything. Not currently implemented.

NPRTPT

Output every NPRTPTth shell of model when dumping model information controlled by NPRT1 and NPRT2 to FMODPT and to FPMOD (pulsation model output).

LPSHLL

If LPSHLL is .TRUE. then additional information about the H burning shell is output to FMODPT, otherwise it is not.

LCONZ0

If LCONZ0 is .TRUE. then addition information about the convection zone is output to FMODPT, otherwise it is not.

LCHEM0

If LCHEM0 is .TRUE. then additional information about the composition is output to FMODPT, otherwise it is not.

LJOUT

If LJOUT is .TRUE. then additional information about rotation is output to FMODPT, otherwise it is not.

LPRTIN

If LPRTIN is .TRUE. then additional information about rotationally induced instabilities is output to FMODPT, otherwise it is not.

NPENV

If NPENV ≥ 1 then output, to FMODPT, information about the envelope and atmosphere every NPENV times the surface integration is performed. If NPENV ≤ 0 do not output anything.

LCORR

If LCORR is .TRUE. then output to FSHORT the shell number and the P, T, R corrections for the shell which has the largest such corrections. This information is useful when trying to figure out why the models are not converging.

NPOINT

If NPOINT ≥ 1 then output every NPOINT shells to FMODPT of the results calculated in the subroutine HPOINT. HPOINT is responsible for redistributing the shells. The output will list the old and the new distributions. If NPOINT ≤ 0 do not output anything.

LRWSH

To prevent the FSHORT file from growing too large in long runs this flag can be set to .TRUE. to rewind the file after every model, thereby, keeping the file small.

Binary model input and output flags

LBNIN, LBNOUT, NBN, NLST

These variables control the use of a special binary formatted input and output of models. The binary format, unlike the standard ASCII text format, is not readable in an editor like vi, EMACS, or EDT. It can only be read by FORTRAN programs running on the computer you used to generate the file. It has two important advantages over the ASCII text format: (1) the numbers stored in the file retain their full precision, and (2) reading and saving data in this format is many times faster than reading and saving the same information in ASCII format. The binary model input/output routines have been implemented as a guide to future enhancements to YREC. Setting LBNOUT to .TRUE. will turn on the binary output routines. If set to .FALSE. no binary output will be performed. This option has been made obsolete by modern computers. Back in the old days it took days of CPU time to reach the tip of the giant branch. Currently, with the ability to produce a hundred models a minute there is less need for this option, unless you are debugging a convergence problem in an advanced phase of evolution.

To read a binary formatted starting model rather than an ASCII text formatted model set LBNIN to .TRUE., otherwise set it to .FALSE. to read the standard ASCII format input model file. Note that the file name is stored in the variable FFSTBN; the file name variable FFIRST is ignored when using a binary formatted starting model.

If LBNOUT is set to .TRUE. then YREC will generate binary model output files for every NBNth model and will generate binary model output files for the last NLST files. If these numbers are less than or equal to zero than no binary models will be output. Consider for example NBN=5 and NLST=2. An output model will saved every NBNth model. Specifically, if the binary store output model file name is designated “storebin.bmod” then the files “storebin.bmod01_0000”, “storebin.mod01_0005”, “storebin.mod01_0010”, etc. will be created and used to save the 0th, 5th, 10th, etc. binary models of the first run (kind card 1). Notice that unlike the STORE.MOD files, only one model is saved in each file, thereby making it easy to resume an evolutionary run with one of the evolved models. In addition, following along with the example, the last 2 models calculated will also be saved in binary format. Specifically, if the binary last output model file name is designated “lastbin.bmod” and the run stops on the 33rd model then the two files “lastbin.bmod01_0033” and “lastbin.bmod01_0032” will contain the binary format form of these two models.

Special interest output flags

LDEBUG

If .TRUE. then debugging information will be output to the FDEBUG file. If .FALSE. no debugging output will be written. Programmers who are modifying YREC can use this flag to turn off and on sections of code they are testing.

LPULSE {F}

If .TRUE. then interior, envelope and atmosphere information will be output for the last model calculated, otherwise no output will be generated. The output is in a form which can be read by jig (Guenther's nonradial stellar oscillation program). In version 1, 2, and 3 (IPVER) output is split into three separate files: FPMOD, FPENV, and FPATM, which contain information about the interior, envelope, and atmosphere, respectively. Every NPRTPTth shell will be output (normally NPRTPT should be set to 1). Implemented in YREC7, version 4 of the output (IPVER=4) sends all of the output to one file (FPMOD), that is the envelope and atmosphere output is appended to the model output. Versions of the pulsation program preceding jig8 cannot read version 4 of the output.

The variables output are RL, FS, B, TL, PL, ESUM, O, QDP, QED, QET, QOD, QOT, DEL, DELA, QCP, RMU, X, and Z. In the IOPENV file RL is not the log of the radius but is the depth measured from the effective temperature surface and in the FPATM file RL is an approximate measure of the thickness of the shell.

This is the easiest way to produce a detailed run of model variables from the center to the top of the atmosphere. A program (gong.f) has been written that will take the PULSE output files and unite them into one file listing nearly all the model data calculated in a human readable form.

IPVER {2}

If IPVER=1 then the pulsation output is done with fewer significant digits (E16.9). If IPVER=2 then the pulsation output is done with more significant digits (E23.16). If IPVER=3 then important header information is also output. Guenther's pulsation code will automatically read these formats. If you are using jig8 of the pulsation program, then set IPVER=4, which produces a single file of output.

LPOUT {F}, POMAX {0.01}, POA {1.0}, POB {10.0}, POC {0.0}

If LPOUT is .TRUE. then dump numbered pulsation output (FPMOD, and if IPVER≤3: FPENV and FPATM) when the track has moved arc length POMAX in HR-diagram. The arc length is defined by:

$$LEN = [POA * \Delta Log(L/L_s)]^2 + [POB * \Delta Log(T_{eff})]^2 + [POC * \Delta Age]^2,$$

where Δ is the model change in the variable.

LAGES, NAGES, AGES(I), I=1, 100

With YREC7 a new pulsation output option is implemented that dumps pulsation model output at user defined ages. Set LAGES=T to turn on this output.. Up to 100 specific ages can be defined. NAGES defines the number of ages, and AGES(I) lists the actual ages, in Gyr. The ages must be monotonically increasing.

YREC will carry out normal evolution, until it gets near one of the listed ages. When it is within one time-step of one of the AGES, YREC will adjust the time-step so that the age of the model hits that age. Once calculated the model details will be output in the pulsation format. In addition, a *.short file will be creating containing a summary of the model's characteristics.

The file names are numbered sequentially, e.g., starmod_0000.pmod, starmod_0001.pmod, etc. The first evolved model in the run is always output as the 0000 model, so one should not set AGES(1)=0.0D0.

LMILNE

If .TRUE. then the Milne invariants will be output to the FMILNE file, otherwise they will not. The file contains, in order: the shell index, the mass, the radius, the pressure, the density, the hydrogen abundance, the luminosity, U, V, W, and index n+1. Every NPRTPTth shell is output. In days predating computers, a model was constructed by fitting surface with interior solutions. The matching of the solutions was cleverly worked out to be most easily accomplished using the Milne invariants.

LISO, LOPT1, LOPT2, LOPT3.

If. .TRUE. then variables needed for isochrone construction will be output to the file FISO. The three flags, LOPT1, LOPT2, and LOPT3 are not currently used but can be used to specify modifications to the output. The specific information output includes a header line that contains, in addition to the three flags, the mass in gm, the initial surface hydrogen mass fraction, the initial surface metal mass fraction, the mixing length parameter, and the bolometric magnitude of the sun. This is followed by a line for each model containing, the model number, the age in years, the luminosity in erg/s, the radius in cm, the effective temperature in K, the gravity in cm/s^2, the central helium mass fraction, and the core helium mass in gm.

File name variables

FLAST

Output file to hold the last converged model. The file is in text format which can be re-read by YREC to resume evolution.

FFIRST

Input file: initial model read in to start run.

FFERMI

Input file containing the Fermi tables.

FDEBUG

Output file used during YREC for debugging.

FTRACK

Output file containing global properties of the evolving model. See description under LTRACK.

FSHORT

Output file containing a summary of the input parameters, a complete listing of all diagnostic information about the runs, and a short summary of each model.

FMILNE

Output file containing the Milne invariants. See description under LMILNE.

FMODPT

Output file containing detailed information about the model at each mass shell.

FSTOR

Output file containing intermediate models which are in a text format which can be re-read by YREC to resume evolution.

FPMOD, FPENV, FPATM

Output file containing information about the interior of the model. Used as an input file by Guenther's pulsation program. If IPVER=4 then the envelope and atmosphere data are also output to this file. Otherwise, the envelope and atmosphere are output to the FPENV and FPATM files.

IMPORTANT: Unlike all other file name variables used in YREC NAMELISTs, these file name variables only define the prefix for the file. The full file name will be created by YREC by

appending .pmod, .penv, and .patm to the FPMOD, FPENV, and FPATM names. For example, if FPMOD='E:\jverne\stars\mymodel' then the interior pulsation output will be sent to the file E:\jverne\stars\mymodel.pmod.

Furthermore, if LAGES=T, then all the pulsation model output file names will be created from FPMOD. In the previous example, with LAGES=T, the file names created for output will be: E:\jverne\stars\mymodel_0000.pmod, E:\jverne\stars\mymodel_0001.pmod, etc., and E:\jverne\stars\mymodel0000.short, E:\jverne\stars\mymodel0001.short, etc. Note that the short file name is built from the FPMOD prefix, not from FSHORT.

FLSTBN
Output files containing full precision binary formatted information about the last model. (This is a dummy logical unit).

FSTOBN
Output files containing full precision binary formatted information about every "stored" model. (This is a dummy logical unit).

FFSTBN
Input file containing a binary formatted initial model.

FDYN
Output file containing dynamo information. No longer implemented.

FSNU
Output file containing neutrino flux information. No longer implemented. Information now is output to ITRACK, ISHORT, and IMODPT.

FSCOMP
Output file containing extended composition information.

IFISO
Output file containing model summary information required to produce isochrones.

FATM
Input file containing a table of surface *T* and *p* from Kurucz model atmospheres.

FMHD1,…, FMHD8
Input files containing the Mihalas, Hummer, Däppen equation of state tables in binary number format.

FOPALE
Input file containing the OPAL eos tables for a specific composition. OPAL provides a program to generate OPAL equation of state tables at user defined Z (solar mixture).

In *PHYSICS* NAMELIST flags LOPALE and LOPAL2001EOS control use of OPAL EOS. FOPALE only read in and used if LOPALE=T. Use the 2001 format files (higher resolution) when LOPAL2001EOS=T.

Opacity variables

LFERG05
If T then use Ferguson low temperature opacities (preferred). This replace Alexander 95 low temperature tables. They are similar to the Alexander 2005 low temperature opacity tables. The tables are 4D meaning that the code automatically reads in and interpolates in Z as well as X, rho and T.

LALEX05
: If T then use Alexander 2005 low temperature opacities. These tables are interpolated linearly between two distinct Z values when diffusion is on.

LALEX95
: If T then use Alexander 95 low temperature opacities. If F do not. The tables can be interpolated in Z.

LKUR90
: If T then use Kurucz 90 low temperature opacities. If F do not. Specific tables for a given Z must be obtained.

LOPAL95
: If T then use OPAL 95 interior opacities. If F do not. Most current tables. The tables are interpolated linearly between two distinct Z values when diffusion is on. Important see LNEWOPAL95.

LNEWOPAL95
: If T (and LOPAL95=T) then use OPAL 95 interior opacities but with full 4D interpolation of X, rho, T **and** Z. This is the preferred opacity.

LOPLA92
: If T then use OPAL 92 interior opacities. If F do not. Specific tables for a given Z must be obtained.

LLAOL89
: If T then use LAOL 89 interior opacities. If F do not. Specific tables for a given Z must be obtained.

LPUREZ {.FALSE.}
: If .TRUE. then read in the pure C and O opacity table (LAOL89) required for late G.B, H.B. and post-H.B. evolution.

ZALEX1
: Specifies the Z for the Alexander opacity tables. YREC will interpolate the Alexander tables to this requested value of Z.

ZALEX2
: Specifies the Z for the Alexander opacity tables for the second set of Z tables. This second set of tables are needed if you are doing Z diffusion calculations (LDIFZ=T).

ZKUR1
: Specifies the Z for the Kurucz opacity tables. YREC will not check to see if the file being read in is at the specified value of Z.

ZKUR2
: Specifies the Z for the Kurucz opacity tables for the second set of Z tables. This second set of tables are needed if you are doing Z diffusion calculations (LDIFZ=T). YREC will not check to see if the file being read in is at the specified value of Z.

ZOPAL951
: Specifies the Z for the OPAL 95 opacity tables. YREC will interpolate the OPAL 95 tables to the requested value of Z.

ZOPAL952
: Specifies the Z for the OPAL 95 opacity tables for the second set of Z tables. This second set of tables are needed if you are doing Z diffusion calculations (LDIFZ=T).

ZOPAL1
Specifies the Z for the OPAL 92 opacity tables. YREC will not check to see if the file being read in is at the specified value of Z.

ZOPAL2
Specifies the Z for the OPAL 92 opacity tables for the second set of Z tables. This second set of tables are needed if you are doing Z diffusion calculations (LDIFZ=T). YRE6 will not check to see if the file being read in is at the specified value of Z.

ZLAOL1
Specifies the Z for the LAOL 89 opacity tables. YREC will not check to see if the file being read in is at the specified value of Z.

ZLAOL2
Specifies the Z for the LAOL 89 opacity tables for the second set of Z tables. This second set of tables are needed if you are doing Z diffusion calculations (LDIFZ=T). YREC will not check to see if the file being read in is at the specified value of Z.

OPECALEX(1)… OPECALEX(7)
Input files containing Alexander opacity tables. These files are interpolated to specified Z.

FKUR, FKUR2
Input files containing Kurucz opacity tables. Specific files for specific Z. FKUR2 is only used if L2Z=T (i.e. if LZRAMP=T or LDIFZ=T).

FLIV95
Input file containing OPAL 95 opacity tables. This file is interpolated to specified Z.

FLLDAT, FOPAL2
Input files containing OPAL 92 opacity tables. Specific files for specific Z. FOPAL2 is only used if L2Z=T.

FLAOL, FLAOL2
Input files containing LAOL 89 opacity tables. Specific files for specific Z. FLAOL2 is only used if L2Z=T.

FPUREZ
Input file containing D.B. Guenther's higher resolution format LOAL opacity for a pure C and O mixture (50% each by number). Used to calculate opacities in the cores of H.B. and post H.B. stars.

TOLALEX, TOLOPAL95
Implemented in YREC7, allows linear extrapolation from the edge of the tables (Alexander and OPAL95, only) by TOLALEX or TOLOPAL95. Amount is in log-normalized temperature and log normalized density.

FDELOP
Input file containing a list of pairs of numbers for Log T and opacity correction factors. Used when LDELOP=T (see LDELOP in "$PHYSICS NAMELIST below).

LMLOSS {.FALSE.}
If T then compute mass loss/gain according to rates set by following parameters:

CMLL, CMLR, CMLM, CMLG, CMLETA, CMLC, ETAML, FRACTML, TINSTLO, TINSTHI
See subroutine massloss.f for details.

$PHYSICS NAMELIST (*.nml2)

Opacity parameters

TOLLAOL {10.0}

When using Guenther's format LAOL opacity tables, cubic spline interpolation is performed inside the table and linear extrapolation is performed outside the table. Linear extrapolation will only be used up to a factor of TOLLAOL (in the log) outside the table. Otherwise and error message will be printed and YREC will stop.

LDELOP {.FALSE.}

If .TRUE. then read in the file FDELOP that lists the pair LogT and opacity correction factor. All opacities will be multiplied by the opacity correction factors. YREC will generate a cubic spline fit of the opacity correction factor as a function of LogT. Whenever an opacity in the model is used it will be multiplied by the correction factor. The value of the correction factor depends on the temperature, as defined in the table.

With LDEOP=T you can modify, in an ad hoc manner, the opacity in a specific temperature range in a star. Simply set the correction factors to 1.0 for Log T=2.0 to 10.0, except in the temperature range where you want a modified opacity.

Optional atmosphere and envelope integration

LENVG, ATMSTP {0.5} [0.002], ENVSTP {0.5} [0.002]

During normal operation of YREC three envelope and atmosphere integrations are performed which define a triangle in the luminosity - effective temperature plane. These three integrations (called the triangle) surround the actual envelope and atmosphere of the interior integration. The three integrations are interpolated at $\tau = 2/3$ to get the surface values of temperature, etc. If LENVG is .TRUE. then an additional atmosphere envelope integration is performed which passes through the interpolated surface values and represents the actual envelope and atmosphere of the interior integration. If LENVG is .FALSE. no such integration is performed. This fourth envelope atmosphere integration is not required to calculate stellar evolutionary sequences but is important if you want output of the actual envelope and atmosphere. The information is output to the IMODPT files.

Because stellar models for pulsation require detailed and precise envelopes and atmospheres the fourth level of integration is performed on the last model when LPULSE is .TRUE. and the results are output to the IOPATM and IOPENV files.

The integration step size for the envelope and atmosphere integrations during this fourth integration are set with ENVSTP and ATMSTP. For pulsation to produce atmospheres and envelopes with approximately 600 shells each set ENVSTP=0.002 and ATMSTP=0.002.

Nuclear burning parameters

LNEUTR {T}

If .TRUE. then include leptonic neutrino emission calculation, otherwise do not. Normally this should be left .TRUE.

LPSEP {F}

If .TRUE. use the old energy generation network of the early PSEP stellar evolution code, otherwise use the more modern implementation. Normally this should be left .FALSE. Note that an even more up to date energy generation network now exists, see LNEWE.

TCUT(1) {6.5}, TCUT(2) {6.5}, TCUT(3) {6.82}, TCUT(4) {7.73}, TCUT(5) {7.5}

To improve the efficiency of YREC nuclear reaction networks are only calculated if the temperature is high enough for the reaction to produce a significant amount of energy. The logarithms of the cutoff temperatures below which specific nuclear reaction networks are not included are specified by TCUT(1…5). They should not be increased in value beyond their default value.

TCUT(1) is the low temperature cutoff for all nuclear reaction chains.

TCUT(2) is the low temperature cutoff for the pp chains.

TCUT(3) is the low temperature cutoff for the CNO cycle.

TCUT(4) is the low temperature cutoff for all the He burning chains.

TCUT(5) is the low temperature cutoff for leptonic neutrino emission.

LNEWE {F}

If .TRUE. use Bahcall (ENGEB) rates to calculate nuclear luminosities; if .FALSE. use the Vandenberg (ENGE) formalism to calculate nuclear luminosities. The Bahcall rates are better and should be used.

LSNU {F}

If LSNU is .TRUE. then print out solar neutrino fluxes to ITRACK, ISHORT, and IMODPT. The neutrino fluxes are calculated in the ENGEB hence LNEWE must also be .TRUE..

LI88

If .TRUE. use latest 1988 Li burning rates; otherwise use older ???? rates.

Equation of State and surface boundary conditions

LOPALE, LOPALE2001EOS, LOPALD {F, F, F}

If LOPALE=T then YREC will use the OPAL eos tables (FOPALE) to determine the relationship between ρ, P, and T at each shell in the model. The OPAL tables are superior to the standard YREC eos routines+Debye-Hückel corrections. They are important when construction models of the Sun. Unfortunately, the routines slow down the execution speed of YREC by almost a factor of 5.

LOPALEOS2001EOS=T then use the higher resolution OPAL EOS tables.

If LOPALD=T, the YREC will use OPAL eos tables to get the derivatives of thermodynamic quantities. This slows down execution a further factor of two and does not affect the model at the level of 7 significant digits. We recommend leaving LOPALD=F.

Recommend setting LOPALE=T and LOPAL2001EOS=T. Set LOPALD=F.

TSCUT {6.0}

For log temperatures greater than TSCUT the gas is assumed to be fully ionized and the Saha equation is not used in the equation of state calculation. For log temperatures below TSCUT-0.5 the Saha equation is used to solve the partially ionized equation of state. In the transition region between these two regions the equation of state interpolates between the partially and fully ionized solutions to obtain a smooth transition. Do not set TSCUT below 5.8.

TENV0{3.0}, TENV1 {9.0}

The envelope integration is confined to the log temperature range TENV0 to TENV1.

TGCUT {6.9}
If the log temperature is less than TGCUT in a convective regions then the mixing length theory used to calculate the temperature gradient, otherwise the temperature gradient is assumed to be adiabatic.

LDH, ETADH0 {-1.0}, ETADH1 {1.0}
If .TRUE. then enhance the standard equation of state calculation by including the Debye Hückel correction, otherwise do not. The Debye Hückel correction, as implemented, corrects the density for the effects of Coulombic interactions between the *fully* ionized ions and the charged plasma. The current formulation is not applicable in regions of partial ionization. Use at own risk.

There are two forms of the DH implemented: one assumes complete degeneracy of the electrons the other assumes no degeneracy. The variables ETADH0 and ETADH1 define where one form or the other is used. When ETA (degeneracy) is less than ETADH0 then assume non degenerate electron gas form of DH correction. When ETA is greater than ETADH1 then assume degenerate electron gas form of DH correction. In between average the two possibilities with a ramp function.

LMHD
If .TRUE. then read in Mihalas, Hummer, and Däppen equation of state tables for the Ross-Aller solar mixture and then use the tables instead of the standard equation of state routines to calculate the density and other thermodynamic variables as a function of pressure and temperature. Currently only the solar Ross-Aller mixture is supported. This must be improved when more complete MHD tables (i.e. more mixtures) are made available.

TENV0 {3.0}, TENV1 {9.0}
The envelope integration is confined to the log temperature range TENV0 to TENV1.

TGCUT {6.9}
If the log temperature is less than TGCUT in a convective regions then the mixing length theory is used to calculate the temperature gradient, otherwise the temperature gradient is assumed to be adiabatic.

Surface boundary condition parameters

KTTAU {0}
If set to 0 then YREC will use the Eddington T-τ relation, if set to 1 then YREC will use the Krishna Swamy T-τ relation. The latter is a fit to the Sun. (To be implemented in the future: a fit to the Harvard Smithsonian Reference Atmosphere.)

LATMTAB {F}
If .TRUE. then interpolate in Kurucz model atmosphere tables (FATM) rather than constructing an Eddington atmosphere.

LDIAZ {F} [obsolete]
If .TRUE. then calculate the physical conditions at the top of the envelope by interpolating the Gustafsson-Bell model atmosphere otherwise use the gray atmosphere integrator. This should always be left .FALSE.

LNEW0 {F}
If .TRUE. then a new envelope triangle is calculated for each model otherwise only calculate a new envelope triangle edge (i.e. flip the triangle) when the model steps out of the triangle. For rotating evolution LNEW0 is forced to .TRUE.

TRIDT {0.01} [1.0D-4], TRIDL (0.08} [8.0D-3], TRITOL (0.0}
TRIDT and TRIDL are the edge lengths in the T-L plane of the envelope triangle. Three separate integrations of the atmosphere and envelope are performed which are expected to surround, in the T-L plane, the outer integration point of the interior solution (the matching point between the

interior solution and the envelope atmosphere solution). The three atmosphere/envelope integrations are interpolated to the value of the interior solution at the matching point. The same interpolation is then performed at the surface to give the surface values of T, P, and L. If the L and T values at the matching point lie outside the atmosphere/envelope integrations then one of the integrations is recalculated at a new T, L value, effectively flipping the triangle (note: sometimes two vertices must be calculated).

Decreasing the size of the triangle reduces the error in the linear interpolation scheme but increases the number of times the triangle must be flipped as the star evolves in the T-L plane.

TRITOL is used to allow the interior solution to lie within TRITOL (fraction) outside of the triangle without forcing a flipping of the triangle. If the solution lies farther than this then the triangle is flipped. To ensure stable solutions you may have to use a non-zero value.

ATMERR {3 .0D-4} [3.0D-5]
ATMERR is the maximum error allowed in the atmosphere integration.

ATMMAX {0.5} [0.05], ATMMIN {0.25} [0.015]
Maximum and minimum, respectively, permitted integration step size for the atmosphere integration, in log τ. Effectively controls the number of shells used in atmosphere integration.

ATMBEG {0.1} [0.015]
Initial step size, in log τ, used in the atmosphere integration.

ATMD0 {1.0D-10}
The density is assumed to be zero at log τ = ATMD0.

ENVERR {3.0D-4} [3.0D-5]
ENVERR is the maximum error allowed in the envelope integration.

ENVMAX {0.5} [0.05], ENVMIN {0.1} [0.015]
Maximum and minimum, respectively, permitted integration step size for the envelope integration, in log P. Effectively controls the number of shells used in envelope integration.

ENVBEG {0.1} [0.015]
Initial step size in log P for envelope integration.

STOLR0 {1.0D-3}, IMAX {11}, NUSE {7}
STOLR0 determines how close in mass the integration needs to be to the fitting mass before using the derivatives to extrapolate to conditions at the fitting mass.

IMAX governs the maximum number of times the derivatives are evaluated across the interval.

NUSE is the number of previous guesses at the value at the end of the step that are used.

Helium Flash

LKUTHE {.FALSE.}
If .TRUE. then several modifications designed to follow the evolution through helium flash will be turned on. Modifications include time-step cutting, omission of point redistribution after each model is calculated, and minor modifications to the output. Note: this flag is useful if you specifically want to turn off the point redistribution routine because it is introducing instabilities into your models.

Rotation parameters

LROT {.FALSE.}
If .TRUE. then YREC goes into rotating evolution mode otherwise standard nonrotating evolution is calculated.

LINSTB {.FALSE.}
If .TRUE. then hydrodynamic instabilities are calculated otherwise they are not.

LJDOT0 {.TRUE .}, ALFA {1.5}, FK {2.46}
If LJDOT0 is .TRUE. a stellar wind based on the Belcher-MacGregor model is calculated and angular momentum is drained form the surface convection zone. ALFA and FK are wind law constants which define the derivative $\partial J/\partial T$ as $\partial J/\partial T$ = FK*OMEGA**(ALFA+1).

WALPCZ {0.0}
Used to enforce a power law rotation curve (Ω = R**WALPCZ) in the outer convective zone. Zero corresponds to solid-body rotation and 2 corresponds to uniform angular momentum per unit mass.

ACFPFT {1.0D-36}
Relative accuracy of equipotential surface determination.

ITFP1 {5}
Number of iterations between calculating ETA2 and R0.

ITFP2 {20}
Number of iterations to calculate R0 and HR.

LWNEW {.FALSE.}
If .TRUE. then set initial angular velocity of model to WNEW (i.e. rigid rotation) otherwise do not.

WNEW {1.2E-7}
Initial angular velocity of model.

DTDIF {1.0D-2}
Maximum chance in Ω at any point for one diffusion time-step.

ITDIF1 {1.0}
Description to be provided.

ITDIF2 {1.0}
Description to be provided.

DTWIND {8.0D-2}
Maximum change in Ω at any point for one model time-step.

DJOK {1.0D-4}
Tolerance for diffusion equation.

CZMIN {1.0D-9}
Minimum depth of convection zone for angular momentum loss.

FUDGEW {1.0}
The diffusion coefficients for angular momentum transport are multiplied by FUDGEW.

FUDGEC {1.0}
The diffusion coefficients for composition transport are multiplied by FUDGEC.

FUDGE1 {1.0}
Multiply diffusion coefficient at base of convection zone by FUDGE0.

FMU {1.0}
Sensitivity of Eddington-Sweet and Goldreich-Schubert-Fricke instabilities to gradients are proportional to FMU.

RCRIT {1000.0}
Critical Reynolds number at which point shear instabilities set in.

FES {1.0}
Factor by which Eddington-Sweet velocity estimates are multiplied.

FGSF {1.0}
Factor by which Goldreich-Schubert-Fricke velocity estimates are multiplied.

FSS {1.0}
Factor by which shear velocity estimates are multiplied.

FESC {1.0}
Diffusion coefficient for Eddington-Sweet instability.

FESSC {1.0}
Diffusion coefficient for secular-shear instability.

FGSFC {1.0}
Diffusion coefficient for Goldreich-Schubert-Fricke instability.

LSMOOTH {F}
Not implemented.

Structure convergence criteria

Note: this section contained errors in YREC6 version of the documentation. They have been corrected.

HTOLER(1,1) {6.0D-5} [6.0D-7]
Convergence criterion for the pressure correction, δ(log P). Converged if δ(log P) < HTOLER(1,1). The model is considered converged if P, T, L, and R corrections are less than HTOLER(I,1), I=1, 4.

HTOLER(2,1) {4.5D-5} [4.5D-7]
Convergence criterion for the temperature correction, δ(log T). Converged if δ(log T) < HTOLER(2,1).

HTOLER(3, 1) {3.0D-5} [3.0D-7]
Convergence criterion for luminosity correction, $\delta R/R_S$. Converged if $\delta R/R_S <$ HTOLER(3, 1).

HTOLER(4, 1) {9.0D-5} [9.0D-7]
Convergence criterion for luminosity correction, $\delta L/L_S$. Converged if $\delta L/L_S <$ HTOLER(4, 1). Note this value is not logarithmic hence must be increased when constructing models of stars of great luminosity. Relax this criterion when evolving intermediate mass stars up the giant branch.

HTOLER(5, 1) (3.0D-5} [3.0D-7]
Convergence tolerance for log P and log R in the finite difference equation solution.

HTOLER(1, 2) {9.0D- 1}
Maximum allowed correction to log P. If the correction exceeds this number than YREC assumes the model has diverged.

HTOLER(2, 2) {5.0D-1}
Maximum allowed correction to log T. If the correction exceeds this number than YREC assumes the model has diverged.

HTOLER(3, 2) {5.0D- 1}
Maximum allowed correction to L/L_S. This number must be set to 500 or greater to do giant branch evolution. If the correction exceeds this number than YREC assumes the model has diverged.

HTOLER(4,2) {2.0}
Maximum allowed correction to log R. If the correction exceeds this number than YREC assumes the model has diverged.

HTOLER(5,2) {2.5D-6} [2.5D-8]
Tolerance on the right hand side of the finite difference equations for T and L (important primarily during He flash).

Distribution and resolution of mass shells

HPTTOL(1) {1.0D-8}
Minimum change in log M between Henyey grid points. Any shell with a mass less than HPTTOL(1) will be removed. Note, for low mass envelopes, decrease this number.

HPTTOL(2) {0.08} [0.02]
Maximum allowed change in log M between Henyey grid points. Primarily controls the number of mass shells in the center of the star and for the entire He core during giant branch evolution. Decrease this number to increase the number of shells.

HPTTOL(3) {5.0D-2}
If the change in X between two adjacent grid points exceeds HPTTOL(3) then the grid points are flagged. YREC will preserve the composition discontinuities at the flagged points (e.g. the possible composition discontinuities at the edges of convection zones are flagged).

HPTTOL(4) {1.0}
If the change in Y between two adjacent grid points exceeds HPTTOL(4) then the grid points are flagged and the composition discontinuity is maintained. Setting HPTTOL(4) = 1.0 turns off flagging for Y discontinuities.

HPTTOL(5) {0.05} [0.01]
Maximum change in log P between adjacent Henyey grid points in the surface convection zone and above the fine mesh zone. This variable governs the number of shells in the surface convection zone excluding the fine mesh zone. A fine mesh zone can be defined around the surface convection zone and is a region were the shell spacing can be adjusted using HPTOL(10). Setting HPTTOL(5,6,10,11) = 0.01 will give about 750 shells in solar model.

HPTTOL(6) {0.02} [0.01]
Maximum change in log L between adjacent Henyey grid points. Primarily used to govern shell spacing in the nuclear burning regions, where L changes. HPTTOL(5,6,10,11) = 0.01 will give about 750 shells in solar model. Recommend value near 0.001 on giant branch.

HPTTOL(7) {1.0}
Maximum change in *X* between adjacent Henyey grid points. Setting HPTTOL(7) = 1.0 turns off this constraint. Must be decreased to 0.005 on giant branch to correctly evolve through zig-zag.

HPTTOL(8) {1.0}
Maximum change in *Z* between adjacent Henyey grid points. Setting HPTTOL(8) = 1.0 turns off this constraint.

HPTTOL(9) {1.0D- 1}
For rotating models HPTTOL(9) is the maximum allowed discontinuity in Ω between adjacent Henyey grid points. If the value is exceeded then the grid points are flagged.

HPTTOL(10) {0.05} [0.01]
Maximum change in log *P* between adjacent Henyey grid points in the fine mesh region. The fine mesh region is a region that the user defines around (see HPTTOL(12)) the base of the surface convection zone within which the grid spacing is separately controlled by HPTTOL(10). HPTTOL(5,6,10,11) = 0.01 will give about 750 shells in solar model.

HPPTOL(11) {0.05} [0.01]
Maximum change in log *P* between adjacent Henyey grid points below the surface convection zone and below the fine mesh region. HPTTOL(5,6,10,11) = 0.01 will give about 750 shells in solar model.

HPPTOL(12) {0.0}
The fine mesh zone extends a distance HPTTOL(12) in log *P* below base of the over shoot region and above the base of the convection zone.

Special composition parameters

LEXCOM {.FALSE.}
If .TRUE. then the abundances of ^{2}H, ^{6}Li, ^{7}Li, and ^{9}Be will be tracked otherwise they will not. Note LEXCOM must match the startup model's LEXCOM flag otherwise YREC will abort.

OPTOL {1.0D-8}
The opacity table will not be interpolated in composition and the nearest table value will be used if the composition lies within OPTOL of the table's composition.

Iteration control

NITER1 {2}
Maximum number of iterations allowed to achieve convergence during first set of corrections.

NITER2 {20} [40]
Maximum number of iterations allowed to achieve convergence during second set of corrections.

NITER3 {0}
Maximum number of iterations allowed to achieve convergence during third set of corrections. LNEWE must be set to .TRUE. See Program Flow section for description of this level of iteration. Set NITER3=10 if used.

The third and fourth levels of iteration solve the composition at the end of the time-step not the beginning as in the first and second levels of iteration. In the first and second levels the nuclear reaction rates of the previous model are used to calculate the new composition. In the third and fourth levels of iteration the nuclear rates of the converged model (from the prior level of iteration) are used to estimate the new model's composition. An improved guess at the nuclear reaction rates

are then calculated based on the first estimated of the new model's composition. The rates and composition are iterated until they converge.

The third and fourth levels of iteration introduce noise in the track near the tip of the giant branch. We advise turning the third and fourth levels off at log $L/L_S > 2.5$.

NITER4 {0}
Maximum number of iterations allowed to achieve convergence during fourth set of corrections. LNEWE must be set to .TRUE. See Program Flow section for description of this level of iteration. Set NITER4=10 if used. See comments in NITER3.

LNEWS {F}
If .TRUE. then use last model's entropy term to calculate initial guess structure for new model, otherwise a zeroed entropy term is used for the initial guess structure. Does not speeds up convergence very much. Regardless, recommend always setting to .TRUE.

FCORR0 {0.8}
After each iteration the corrections or some fraction of them are added to the previous solution. Initially FCORR0 times the corrections are added.

FCORRI {0.1}
After the initial iteration the factor by which the corrections are multiplied is given as FCORR0 + (n-1) * FCORRI, where n is the number of iterations. The factor is limited to a maximum of 1.0.

IHPOINT {1}
Call subroutine HPOINT every IHPOINT'th model. Normally this is set to one. On the G.B. setting IHPOINT to 10 adds some stability to the calculation.

LINHPNT {F}
Normally oscillatory splines are used HPOINT to determine the values of variables in newly places shells. If LINHPNT is set to .TRUE. then use simple linear interpolation. Has little known affect.

Time-step size

ATIME(l) {0.001}
If *X* falls below ATIME(l) in the center of the model then the model is assumed to have a H burning shell.

ATIME(2) {0.02}
Maximum change in central *X* allowed between adjacent, evolving models. If this number is exceeded then the time-step is considered to be too large, i.e. too much H was burnt between adjacent models. This number governs main-sequence time-steps.

ATIME(3) {0.5}
Maximum fraction change in central *X* allowed between adjacent, evolving models. This number governs the time-steps during H exhaustion where ATIME(2) has little effect.

ATIME(4) {0.02}
Maximum change in central *Y* allowed between adjacent, evolving models. This governs time-steps during core He burning, i.e. on the horizontal branch.

ATIME(5) {0.3}
Maximum fraction change in central *Y* allowed between adjacent, evolving models. This number governs the time-steps during He exhaustion where ATIME(4) has little effect.

ATIME(6) {0.0015}
Maximum mass allowed to bum in a H shell or double shell burning region between adjacent, evolving models. This number governs time-steps on the upper giant-branch and the horizontal branch. Stable and accurate evolution on giant branch is achieved with 0.00060.

ATIME(7) {0.1}
Maximum allowed depletion in X at the midpoint of the H-shell between adjacent, evolving models. This governs time-stepping early phases of giant-branch evolution (and horizontal branch evolution?).

ATIME(8) {0.02}
Maximum allowed change in log P from one model to the next.

ATIME(9) {0.4}
Maximum allowed change in log T from one model to the next.

ATIME(10) {0.02}
Maximum allowed change in log R from one model to the next.

ATIME(10) {0.02}
Maximum allowed change in log L from one model to the next.

LPTIME {.FALSE.}
Implement time-step criterion based on the changes in the physical variables. This is required during pre-main-sequence evolution but quickly fails during and after main-sequence turn-off.

Time-step cutting options

Normally when the corrections exceed the maximum amount or YREC has performed the maximum number of iterations and not converged, YREC will stop. Sometimes the problem is a too long time-step. Time-step cutting allows the program to backup one model and try again with a smaller time-step.

LKUT {.FALSE.}
If .TRUE. then time-step cutting is enabled otherwise is not. A bug currently exists in this procedure.

NKUT {5}
The maximum number of times time-step cutting will be attempted before the program gives up and quits.

FKUT {2.0}
The factor that the time-step is reduced with each attempt.

Minor startup model modifications

LNEWCP {.FALSE.}
If LNEWCP is .TRUE. then rescale the abundance of one element otherwise do not.

ANEWCP {three blank characters}
A three letter character variable representing the element to be rescaled. The following abbreviations are used, HE3, C12, C13, N14, N15, 016, 017, 018, H2 (blank character at end), LI6, LI7, BE9.

ATMP {REL}
A three letter character variable that tells YREC whether the new abundance is the absolute abundance (ABS) or is on the standard spectroscopic scale relative to H (REL). REL assumes log H = 12.0.

XNEWCP
The desired abundance of the element ANEWCP.

Deep mixing and overshoot option

LSEMIC {.FALSE.}
If .TRUE. turn on semiconvection routine.

DPENV {1.0}
Artificially mix from the surface to DPENV = $1 - M_{OV}/M_*$.

LOVSTC {.FALSE.}
If .TRUE. then carry out overshoot above core convection zone.

ALPHAC {0.0}
Depth of core overshoot layer (in units of pressure scale height).

LOVSTE {.FALSE.}
If .TRUE. then carry out overshoot below surface convection zone.

ALPHAE {0.0}
Depth of overshoot below surface convection zone.

LOVSTM {.FALSE.}
If .TRUE. then carry out overshoot below and above middle convection zones.

ALPHAM {1.0}
Depth of overshoot below and above middle convection zones.

Nonstandard core mixing

LNSTDMX
If .TRUE. then YREC will force mixing in the model before each evolution step. The array PNSTDMX is used to define the range of mixing. The temperature gradient is not directly altered.

PNSTDMX(1)… PNSTDMX(10)
If PNSTDMX(1) = 1 then mix from center of the model to mass fraction PNSTDMX(2) at the beginning of every evolutionary time-step.

If PNSTDMX(1) = 2 the mix from center of the model to mass fraction PNSTDMX(2) when the age of the stellar model just passes the age PNSTDMX(3). PNSTDMX(4) us used to keep track of when the mixing event occurs; it must be set to 0.0.

If PNSTDMX(1) = 3 then mix from mass fraction PNSTDMX(2) to mass fraction PNSTDMX(3) at the beginning of every evolutionary time-step.

Extend inner most shell

LCORE {.FALSE.}
If .TRUE. after initial model is read in add shells inward. This does not work very well.

FCORE {1.0}
Extend the inner most shell's mass to HS(1)/FCORE where HS(1) is previous inner most shell's mass.

MCORE {1}
Add MCORE shells between HS(1) and HS(1)/FCORE.

Gravitational Settling Diffusion

LDIFY {.FALSE.}
If .TRUE. then turn on calculation of gravitational settling diffusion for He.

LDIFZ {.FALSE.}
If .TRUE. then turn on calculation of gravitational settling diffusion of Z. A second opacity table is generated or read in.

LTHOUL
If T then use Thoul, Bahcall, and Loeb diffusion coef.

LTHOULFIT
If T then use a fit to the Thoul et al. diffusion coef. If F then use Thoul et al. subroutines calculate diffusion coeff.

ILAMBDA
1-assign all values of coulomb logs to ln(lambda) = 2.2D0. 2-assign all values of Coulomb logs to ln(lambda) from Noerdlinger's formula. 3-assign all values of Coulomb logs to ln(lambda) from Loeb's formula. 4-calculate table of coulomb logs explicitly.

FGRY {1.0}
Factor by which the helium diffusion coefficients are multiplied.

FGRZ {1.0}
Factor by which the Z diffusion coefficients are multiplied.

GRTOL {1.0D-8}
Convergence tolerance for gravitational settling calculation.

NITER_GS {10}
Maximum number of iterations to converge gravitational settling calculation.

DT_GS {0.1}
The fraction DT_GS of settling timescale is used as the minimum time-step in gravitational settling of He.

XMIN {1.0D-3}
Only calculate gravitational settling if the H abundance is greater than XMIN

YMIN {1.0D-3}
Only calculate gravitational settling if the He abundance is greater than YMIN

Nonstandard mixing length

MIXKND {0}
If 0 use the standard YREC implementation of Böhm Vitense mixing length theory.

If 1 use the Stothers and Chin (Ap.J. 440, 297) formulation of mixing length with coefficients that reproduce the Böhm-Vitense mixing length theory.

If 2 use the Stothers and Chin formulation of mixing length with coefficients that reproduce Canuto & Mazzitelli mixing length theory.

LHPZ {T}
Used when MIXKIND .NE. 0.

If T then the mixing length = (mixing length parameter) (pressure scale height).

If F then the mixing length = (mixing length parameter) (depth)

LQEQ1
Used when MIXKIND .NE. 0.

If T then set the Q parameter of Canuto & Mazzitelli (see Stothers and Chin) equal to one.

If F then set the Q parameter of Canuto & Mazzitelli (see Stothers and Chin) equal to correct thermodynamic value.

MIXVAR {0}
Used when MIXKIND .NE. 0

If 0 then use constant mixing length parameter.

If 1 then take α = HPBETA(1)*ln(P/HPBETA(3))+HPBETA(2), where HPBETA(1) is slope of the α versus ln P plot in Y.C. Kim's 3D compressible simulations of convection, PBETA(2) is α at surface, and HPBETA(3) is surface pressure (cgs).

NOTE that MIXVAR=1 makes sense only of LHPZ=.TRUE. and MIXKIND=1.

If 2,… refer to description in "altmix.f" which impliments a variety of alternatives to the standard mixing length theory.

HPBETA(1...10)
Used to implement variable mixing length parameter, MIXVAR .NE. 0.

LDPREE {F}
If .TRUE. then invoke the Deupree-Varner variable mixing length to pressure scale height theory, otherwise do not. Note the default is not to use this special form of the mixing length theory.

Miscellaneous

CMIN {1.0D-20}
A constant used to check against division by zero in CHECKC and NKEMCOM

ABSTOL {1.0D-5}
Absolute tolerance used in NKEMCOM

RELTOL {1.0D-4}
Relative tolerance used in NKEMCOM

KEMMAX {50}
Maximum number of iteration in NKEMCOM