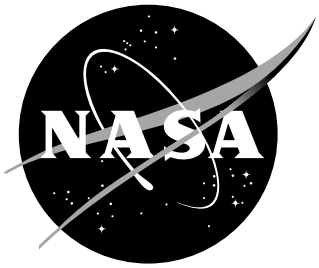
NASA/TM–2010–216836



LAURA Users Manual: 5.3-48528

Alireza Mazaheri
Analytical Mechanics Associates Inc., Hampton, Virginia

Peter A. Gnoffo, Christopher O. Johnston, and Bil Kleb
Langley Research Center, Hampton, Virginia

August 2010

NASA STI Program ... in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collection of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

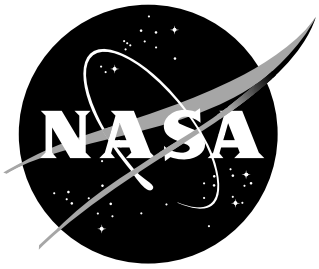
- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.
- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.
- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.
- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.
- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include creating custom thesauri, building customized databases, and organizing and publishing research results.

For more information about the NASA STI Program, see the following:

- Access the NASA STI program home page at ***http://www.sti.nasa.gov***
- E-mail your question via the Internet to help@sti.nasa.gov
- Fax your question to the NASA STI Help Desk at 443-757-5803
- Phone the NASA STI Help Desk at 443-757-5802
- Write to:
  NASA STI Help Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM–2010–216836



LAURA Users Manual: 5.3-48528

Alireza Mazaheri
Analytical Mechanics Associates Inc., Hampton, Virginia

Peter A. Gnoffo, Christopher O. Johnston, and Bil Kleb
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virbinia 23681-2199

August 2010

Available from:

NASA Center for AeroSpace Information
7115 Standard Drive
Hanover, MD 21076-1320
443-757-5802

Abstract

This users manual provides in-depth information concerning installation and execution of Laura, version 5. Laura is a structured, multi-block, computational aerothermodynamic simulation code. Version 5 represents a major refactoring of the original Fortran 77 Laura code toward a modular structure afforded by Fortran 95. The refactoring improved usability and maintainability by eliminating the requirement for problem-dependent re-compilations, providing more intuitive distribution of functionality, and simplifying interfaces required for multi-physics coupling. As a result, Laura now shares gas-physics modules, MPI modules, and other low-level modules with the Fun3D unstructured-grid code. In addition to internal refactoring, several new features and capabilities have been added, e.g., a GNU-standard installation process, parallel load balancing, automatic trajectory point sequencing, free-energy minimization, and coupled ablation and flowfield radiation.

Contents

1 Introduction

The users manual consists of seven sections. Section 2 gives an overview of new features, capabilities, and bug fixes. System requirements and installation are covered in Section 3, followed by code execution instructions in Section 4. Section 5 presents input files, their formats, and detailed information on their contents while Section 6 covers output files. Ancillary utilities are explained in Section 7, and the last section, Section 8, presents illustrative example cases.

2 New in This Version

LAURA v5.3 offers several new enhancements and bug fixes since the previous released version, v5.2 [1]:

- Major Enhancements
  - Enhanced multi-processor per block algorithm so that the total memory no longer scales linearly with the number of processors per block. Now the memory will always be less than twice the memory required by 1 processor per block.
  - Multi-species reacting characteristic boundary condition (see Section 5.3 on page 15) for Hypersonic/Supersonic Retro Propulsion (H/SRP) and Reaction Control System (RCS) applications.
  - Various single and multi-equation turbulence models.
  - Second order time accurate simulation.
- Minor Enhancements
  - An enhanced 1-D grid alignment algorithm and an introduction of a grid limiter for further fine tuning—see Section 5.4.6 on page 25.
  - Wall cell Reynolds number output in `laura_blayer.dat` file.
  - Option to output all the computed variables into `laura.q` file—see Section 5.4.9 on page 26.
  - Allow `thermodynamic.f90` to allow every processor rather than only `lmpi_master`. This removes unnecessary load on `lmpi_master`, which was the major run-time hang up of some large cases, during information broadcasting calls.
  - Option to write `laura_blayer.dat` file with an independent frequency—see Section 5.4.9 on page 26.
  - Use global time stepping if CFL number is negative.
  - Option to monitor surface properties at various time intervals for time accurate simulation.
  - Option to limit individual reaction rates.
  - Added utilities for grid/solution sequencing—see Sections 7.2 on page 60 and 7.8 on page 62.
- Bug Fixes
  - Double entry for $NO^+ + N \Leftrightarrow O^+ + N_2$ in `kinetics_data`.
  - Electron impact ionization.

- Blks array with an index off through the entire subroutine `atime`.
- Incorrectly updating the `blk_{i,j,k}{min,max}` in the `fill_solution_slice` and `update_sub_block` subroutines.
- Incorrectly inserting `dr_{i,j,k}{min,max}` into `blk_{i,j,k}{min, max}` within the `if (dbl_k)` section.
- Runtime failure when using the Lahey compiler due to a perturbed shock adaptation path.
- Boundary layer detection error for non k-surfaces.
- Doubling k points when sweep direction was not in k-direction.
- Undefined `istop` in `main.f90` .
- Gracefully stopping the code if something goes wrong within `thermodynamic.f90` routine.

3 Installation

LAURA requires a Fortran 95 compiler, and if parallel processing is desired, a Message Passing Interface (MPI) implementation.[1] Some optional utilities require Ruby.[2] The installation and subsequent execution of LAURA assumes a Unix-like operating system or compatibility layer.[3] After the code is unpacked from the LAURA release tarball,

```
% tar zxf laura-5.3-Z.tar.gz  (unpack gzipped tarball)
% cd laura-5.3-Z
```

where `Z` is a revision track number. LAURA is installed via GNU build system,[4] which entails executing a sequence of four commands: `configure`, `make`, `make check`,[5] and `make install`.[6]

3.1 Sequential installation

To configure, compile, test, and install a sequential version of LAURA for use with a single processor, first make a subdirectory of `laura-5.3-Z` to store the configuration. For example,

```
% mkdir g95-seq
% cd g95-seq
```

if using the g95 Fortran compiler;[7] and then proceed with the typical GNU build sequence,

```
% ../configure FC=g95 --prefix=$PWD
% make
% make check
% make install
```

Note that `configure`'s `--prefix` option specifies the root directory for installing build artifacts—the default is `/usr/local`. In this example, it is set to the current working directory, `g95-seq` so executables will be installed in `g95-seq/bin` and data files will be copied to `g95-seq/share/laura` and `g95-seq/share/physics_modules` directories.

To use LAURA and associated utilities, set your search path to include `$PWD/bin`, e.g.,

```
setenv PATH ${PWD}/bin:$PATH  (for csh)
export PATH=${PWD}/bin:$PATH  (for sh)
```

[1] For example, OpenMPI or MPICH.

[2] See ruby-lang.org.

[3] For non-Unix-like systems, compatibility layers are available from mingw.org (minimal) and cygwin.com (maximal).

[4] See gnu.org/software/autoconf/.

[5] The `make check` command is optional. It will attempt to run small test cases.

[6] The `make install` command may require administrator privileges depending on your installation location.

[7] g95.org

3.2 MPI Installation

An MPI-enabled installation (to allow multiple processors) is similar to the sequential installation except the configuration command has the `--with-mpi` option instead of the `FC` Fortran compiler variable, e.g.,

```
% ../configure --prefix=$PWD \
               --with-mpi=/usr/local/pkgs/ompi_1.2.8-intel_11.0-028
```

Another difference is that an MPI-enable configuration will produce an executable named `laura_mpi` instead of `laura`.

In either case, `config.log` contains a record of the `configuration` command used, and `configure`'s `--help` option details all available configuration options.

4 Execution

The following steps outline a typical simulation cycle.[8]

Step 1. To start LAURA, a PLOT3D structured grid file is needed — see Section 5.2 on page 14 for more info. You may externally generate a grid using grid generation packages, such as Gridgen™, Grid*Pro*, and so forth, or use LAURA's interactive `self_start` utility to generate a single-block structured grid for simple families of 2D, axisymmetric and 3D blunt bodies—see Section 7.9 on page 62.

Step 1a. Using `self_start`. To use `self_start` to generate a single-block grid, simply execute this interactive utility, e.g.,

```
% self_start
```

and answer all the questions. After a successful execution, this utility will have generated the following files:

```
assign_tasks  laura_bound_data
laura.g       laura_namelist_data  self_start.log
```

Examine the grid, `laura.g`, and proceed to Step 2.

Step 1b. External Grid Generation. Generate a single- or multi-block structured grid with the following rules:

i. Right-handed grid coordinates

ii. Longitudinal axis of the body aligned with the x-axis, oriented nose-to-tail

and write the grid coordinates into a PLOT3D file, `laura.g`. Run the interactive `bounds` utility (see Section 7.1 on page 60) and answer all the questions regarding the grid block topology:

```
% bounds
```

This utility will automatically generate `laura_bound_data`, the connectivity file.

Step 2. If you did not use `self_start`, create `assign_tasks` (see Section 7.7 on page 62 for more info) and a `laura_namelist_data` file or copy the sample file from the `[install_prefix]/share/laura` directory, where `[install_prefix]` is the installation prefix specified when LAURA was installed. Edit this file for your case—see Section 5.4 on page 17 for more detail.

[8] See Section 8 on page 64 for complete worked examples and Appendix A on page 75 for how to restart cases run with versions of LAURA prior to version 5.

Step 3. Create a `tdata` file (see Section 5.5 on page 36) to define the gas model condition for your specific simulation.[9]

Step 4. Run LAURA,

```
% laura
```

or

```
% mpirun -np [#] laura_mpi
```

where `#` is the number of available processors. By the end of this step, the following files will have been generated:

```
laura_conv.out      laura.g.fvbnd
laura_new.g         laura_new.rst
laura_surface.q     laura.q
laura_surface.nam   laura.nam
```

Examine these files before proceeding to the next step.[10]

Step 5. Change `irest` flag in the `laura_namelist_data` (see Section 5.4 on page 17) from `0` to `1`, and copy the new generated grid and solution files to `laura.g` and `laura.rst` files; i.e.,

```
% cp laura_new.g laura.g
% cp laura_new.rst laura.rst
```

Step 6. Repeat the previous two steps until iterative convergence.

[9] A sample `tdata` is available in the `[install_prefix]/share/physics_modules` installation directory. The other datafiles that reside in this directory, e.g., `kinetics_data`, `species_thermo_data`, `species_transp_data`, and `species_transp_data_0`, may also be copied and tailored to suit a different thermodynamic model, curve-fit data, or thermochemical reactions are needed. See Section 5.5 on page 36 for more detail.

[10] See Section 6 on page 55 for complete description of laura output files.

5 Input Files

Nominally, LAURA requires five input files as shown in the upper section of Table 1. Depending on the simulation requirements, however, other files may also be necessary and are shown in the second section of Table 1. All files are plain ASCII text unless otherwise noted.

Table 1: LAURA input files.

Filename	**Content**
REQUIRED FILES:	
`assign_tasks`	Sweep and relaxation directions
`laura.g`*	PLOT3D grid
`laura_bound_data`	Grid block face boundary conditions
`laura_namelist_data`	Simulation configuration
`tdata`	Gas model
SIMULATION DEPENDENT FILES:	
`hara_namelist_data`	Radiation mechanisms
`jdata`	Jet chamber conditions
`kinetic_data`	Specie reactants and products
`laura.rst`†	Flowfield solution for restart
`laura.trn`	Transition location and length
`laura_trajectory_data`	Trajectory points
`laura_vis_data`	Viscous term treatment
`species_thermo_data`	Specie thermodynamics
`species_transp_data`	Collision cross-sections
`species_transp_data0`	High-order collision cross-sections
`surface_property_data`	Thermochemical surface properties

* Fortran unformatted binary, 3-D whole, multiblock PLOT3D.
† Fortran unformatted binary.

The following subsections describe all input files in detail, beginning with the nominally required files and then proceeding alphabetically as shown in the table.

5.1 `assign_tasks`

This file defines sweep and relaxation directions for each grid block. Each line, corresponding to each grid block, has five integers[11] that are separated by at

[11] The code will not read data beyond the fifth column.

least one space. These integers correspond to `nbk`, `mbk`, `mbr`, `lstrt`, and `lstop` where

`nbk`

Block number.

`mbk`

Sweep direction. The assigned value can be either `1`, `2`, or `3`, corresponding to i-, j-, or k-direction, respectively.

`mbr`

The line relaxation direction. Note: must be different than the sweep direction.

Options are:

`0`: Point-implicit, i.e., no line-relaxation

`1`: Line-implicit along i coordinate

`2`: Line-implicit along j coordinate

`3`: Line-implicit along k coordinate

`lstrt`

The starting grid index in the sweep direction. Typically `1` .

`lstop`

The ending grid index in the sweep direction. Typically `0`, which is shorthand for the maximum index.

When starting a new simulation where the k-coordinate runs from the vehicle surface to the freestream boundary, sweeping in the k-coordinate and solving point-implicitly is recommended, i.e. `mbk = 3` and `mbr = 0`. After the shock has stabilized, switch to streamwise sweeps and solve line-implicitly along the k-coordinate, i.e. `mbk = 1` and `mbr = 3`.

5.1.1 Example 1: Multiple Blocks per CPU or Vice-versa

Suppose the grid has 2 blocks, but the number of available processors is not the same as the number of grid blocks. The user does not need to change any of the input files and can simply specify the number of processors available, e.g.,

```
% mpirun -np [#] laura_mpi
```

where `#` is the number of available processors. LAURA automatically assigns processors to blocks such that each processor receives approximately same number of grid cells.

5.1.2 Example 2: Deactivating Grid Blocks

Suppose the grid has 50 blocks, but only blocks 20–25 need to be updated. In this case, `assign_tasks` would contain only the active blocks, e.g.,

```
20  3   0   1     0     1
21  3   0   1     0     2
22  3   0   1     0     3
23  3   0   1     0     4
24  3   0   1     0     5
25  3   0   1     0     6
nbk mbk mbr lstrt lstop dummy
```

5.2 laura.g

This file is a multi-block, 3D-whole PLOT3D file in Fortran unformatted binary format with double-precision reals. For convenience, here is a sample of the Fortran 95 code that LAURA uses to read this file:

```
open ( 25, file='laura.g', form='unformatted' )
read(25) nblocks
...allocate i,j,kblk(nb) and grid memory...
read(25) (iblk(nb),jblk(nb),kblk(nb),nb=1,nblocks)
do nb = 1, nblocks
  ix1 = iblk(nb) ; jx1 = jblk(nb) ; kx1 = kblk(nb)
  ...allocate grid(nb)%x,y,z memory...
  read(25) (((grid(nb)%x(i,j,k),i=1,ix1),j=1,jx1),k=1,kx1), &
           (((grid(nb)%y(i,j,k),i=1,ix1),j=1,jx1),k=1,kx1), &
           (((grid(nb)%z(i,j,k),i=1,ix1),j=1,jx1),k=1,kx1)
end do
```

A file of this format, but named `laura_new.g`,[12] is generated by `Laura` at the end of a successful run. This file is required and must have a right-handed coordinate system. Figure 1 shows the default `laura` coordinate orientation. `Laura` does not require a specific coordinate or grid orientation but the angle-of-attack definition is predefined — see Section 5.4.3 on page 22 for more info.

5.3 laura_bound_data

Grid block face boundary types are defined in `laura_bound_data` where each line corresponds to a grid block and contains six integers, one for each of the six faces: i_{min}, i_{max}, j_{min}, j_{max}, k_{min}, and k_{max}. Each integer specifies either

[12] When running a trajectory sequence, the file will be named `laura_####.g` where `####` is the trajectory point index.

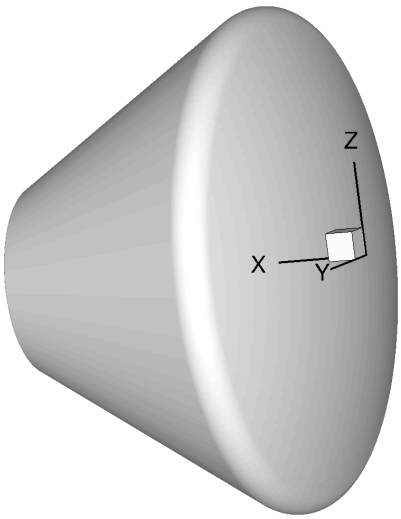


Figure 1: Default LAURA coordinate system orientation.

a physical boundary condition or block-to-block interfaces. An illustrative example is analyzed toward the end of this section.

This file is required and is generated automatically for grids created by LAURA's `self_start` utility—see Section 7.9 on page 62. This file can also be created by using LAURA's interactive utility, `bounds`, by answering questions for each block.

Valid face types are as follows:

`-9,...,0`: Solid surface boundary. Up to ten different solid surface boundaries may be specified. Thermochemical properties of solid surfaces that are different than type `0`, which are specified in `laura_namelist_data`, are defined in `surface_property_data` file—see Section 5.16 on page 52.

`1`: Outflow boundary (extrapolation).

`2`: Symmetry boundary across y = constant.

`3`: Farfield/Freestream boundary.

`4`: Symmetry boundary across x = constant or z = constant.

`5`: Reflection boundary across $j = 1$ symmetry (valid for axi-symmetric and/or 2D grids).

`6`: Venting boundary. (See Section 5.4.18 on page 36 for more details.)

`7`: Reflection boundary across i face singularity with periodic j boundary.

`8`: Characteristic (also known as subsonic) boundary condition.

`>1000000`: This seven digit boundary number defines block-to-block face connectivity. The first digit is always `1`. The next three digits identifies the block number that is shared with the current block. The 5^{th} digit defines which i, j, or k face of the neighboring block is shared where `1` corresponds to i_{min}, `2` corresponds to i_{max}, and so forth. The last two digits identify the relation of the remaining two indices: The 6^{th} digit can be either `1`, `2`, `3`, or `4` where values of `1` or `3` mean the first index of the host face is in the same direction as the first or second index of the neighboring face, respectively, and values of `2` or `4` mean the adjoining indices are in the opposite direction. The last digit can be either a `1` or `2` and indicates whether the second indices of the host and neighboring faces are in the same direction or they are in the opposite direction, respectively.

For example, consider a block with the following `laura_bound_data` boundary condition numbers:

```
1006421  1002111     2   1005121     0       3
```

The first integer, `1006421`, shows that i_{min} of this block is shared with j_{max} (fifth digit) of block 6 (the first three digits after `1`). The first and second indices of the host face are j and k, respectively. Because the j index is connected to i, the first and second indices of the neighboring face are i and k, respectively, The `2` in the 6^{th} digit shows that the j index of the host face is in opposite direction of the i index of the neighboring face. The last digit, `1`, indicates that k indices of the host and and neighboring faces are along the same direction. This configuration is illustrated in Figure 2.

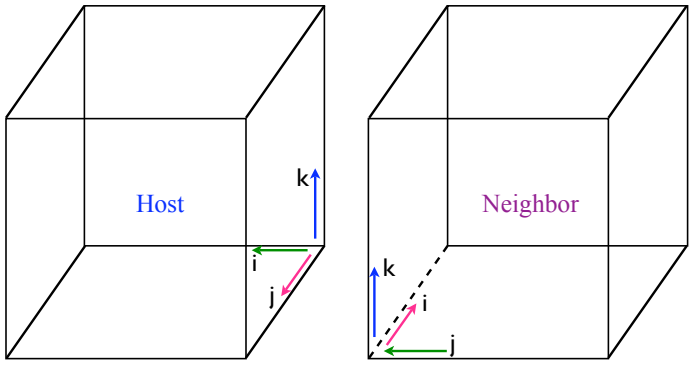


Figure 2: Illustration used for block connectivity example.

The second and fourth boundary-type integers can be explained similarly. The third boundary-type integer (corresponding to the j_{min} face) specifies a y-constant symmetry plane; the fifth boundary-type integer (corresponding to the k_{min} face) specifies is a solid-wall boundary condition; and the last digit (corresponding to the k_{max} face) specifies a freestream boundary.

5.4 laura_namelist_data

Simulation configuration is specified through `laura_namelist_data` and is required. This file is read as a Fortran 95 namelist and has the form,

```
 &laura_namelist
  velocity_ref = 5000.0 ! Free stream velocity, m/s
  density_ref  = 0.023  ! reference density, kg/s
  tref         = 250.0  ! Free stream temperature, K
  alpha        = 25.0   ! Angle-of-attack in xz plane, degrees
[ variable     = value  ! Optional comment ]
 /
```

where `variable` and their possible `value`s are described in the following sections, which are grouped according to farfield/freestream and aerodynamic coefficient reference quantities; thermochemical nonequilibrium flags; molecular transport flags; turbulent transport models flags; numerical parameters; grid adaptation, alignment, and doubling parameters; radiation and ablation flags; grid-file description; venting boundary condition flags; and solid surface boundary condition flags. Note that for all but the parameters shown in the above example, reasonable defaults have been chosen and only those parameters that differ from the defaults need to be specified.

Detailed description of parameters and/or flags with their units and default values is presented under each of the above categories. The order of these parameters is arbitrary but is given here alphabetically for better readability.

5.4.1 Ablation Flags

`ablation_option`

An integer that specifies whether the pyrolysis ablation rate and wall temperature are computed in addition to the char ablation rate. This option only effects cases with `bprime_flag` equal to 0 or 1.

Options are:

- 0: The pyrolysis ablation rate and wall temperature are computed, in addition to the char ablation rate, assuming steady-state ablation.
- 1: The pyrolysis ablation rate and wall temperature are held constant (they are set to the values present in `ablation_from_laura.m`) while the char ablation rate is computed.

`ablation_verbose`

A logical flag to print out developer focused info on convergence of ablation. Default: `.true.`

`blowing_model_0`

Character indicator for ablation model specification. Default: `'none'`

Options are:

`'FIAT'`

This `'FIAT'`[13] option computes the blowing rate and surface temperature as a function of heating rate, pressure, and ablator elemental mass fractions. The user must specify the elemental mass fractions for char and pyrolysis gas or default of 100% carbon will be employed.

`'none'`

No ablation-flowfield coupling.

`'specified'`

This option specifies a blowing or suction rate as a function of pressure (see `mdot_pressure_0`). The user must specify the elemental mass fractions for pyrolysis gas or default of 100% carbon will be employed. Any specification for elemental mass fraction of char is ignored in this option.

`'equil_char_quasi_steady'`

This option solves the equilibrium surface ablation problem. The energy balance, elemental mass balance, and char equilibrium constraint are solved to obtain the char ablation rate, wall temperature, and elemental composition at the surface. Along with the pressure from the normal momentum equation, these values define the equilibrium species composition at the wall. The user must specify the surface temperature type to be `surface energy balance` — see Section 5.4.12 on page 29 for more info.

Presently, this model does not include an in-depth material response computation, which would provide the pyrolysis ablation rate and conductive heat flux into the solid material. These two values are required for solving the previously mentioned surface equations. To approximate these values, the steady-state ablation assumption is made, which specifies that the pyrolysis ablation rate is proportional to the char ablation rate and the in-depth conduction is proportional to the enthalpy at the surface.

In the present framework of steady-state ablation, the ablation material is completely defined by `CHONSi_frac_pyrolysis_0` (thus

[13]FIAT is a stand alone software and is needed if `blowing_model_0='FIAT'`. Request for access to FIAT can be made to NASA Ames Research Center.

CHONSi_frac_char_0 is ignored in this option). These are defined below. Note that the computed ablation rate is the sum of the pyrolysis and char components. (See Sections 8.3 on page 70 and 8.4 on page 72 for recommended procedure for an unspecified ablation computation.)

'quasi_steady'

This option specifies a quasi-steady ablation rate as a function of local pressure, heating rate, and temperature. The sublimation temperature and heat of ablation are specified by the user for a given ablator as a function of pressure. (See t_sublimation_0 and h_ablation_0.) If the surface temperature is less than the sublimation temperature a zero blowing rate is defined. Otherwise the blowing rate is given by $\dot{m} = q/\Delta H_{abl}$ with appropriate nondimensionalization employed before use. The user must specify the elemental mass fractions for pyrolysis gas or default of 100% carbon will be employed. Any specification for elemental mass fraction of char is ignored in this option. The user must specify the surface temperature type to be surface energy balance — see Section 5.4.12 on page 29 for more info.

bprime_flag

An integer defining if the b-prime approach is applied. Applicable only for blowing_model_0 = 'equil_char_quasi_steady'. See Section 5.4.1 on the preceding page for more details of its application. Default: 1

Options are:

0: Do not use bprime approach, and instead use a rigorous diffusion model. This option is consistent with the "Fully-Coupled" approach defined in Ref. [2].

1: Use b-prime approach. This option is consistent with the "Partially-Coupled" approach defined in Ref. [2].

2: Hold the ablation rate and wall temperature constant from the restart file, while applying the rigorous diffusion model (thus, the surface energy balance and char equilibrium constraint are not satisfied). This option is sometimes useful when transitioning from a bprime_flag = 1 computation to a bprime_flag = 0 computation.

char_density_0

Density of the char, kg/m^3. Default: 256.29536, which is for the heritage AVCOAT.

CHONSi_frac_char_0

This rank 1 vector of extent 5 sets elemental mass fraction of C, H, O, N, and Si species from char. Elemental mass fractions must be in this order and the sum of elemental mass fractions must equal 1.0 . Default: `CHONSi_frac_char_0 = 1.0, 0.0, 0.0, 0.0, 0.0`

CHONSi_frac_pyrolysis_0

This rank 1 vector of extent 5 sets the elemental mass fractions of pyrolysis gas species, which are C, H, O, N, and Si. Elemental mass fractions must be in this order and the sum of elemental mass fractions must be 1. Default is Graphite:
`CHONSi_frac_pyrolysis_0 = 1.0, 0.0, 0.0, 0.0, 0.0`

compute_mdot_initial

An integer defining if the ablation rates are computed before the first flowfield iteration.

Options are:

0: Applies the ablation rates and wall temperatures present in the `ablation_from_laura.m` file.

1: Computes the ablation rates and wall temperatures before the first flowfield iteration.

freq_wall

For `bprime_flag = 1`, it is an integer defining frequency of update to ablation wall boundary conditions. For `bprime_flag = 0`, it is an integer defining frequency of update to the surface energy balance solution, which defines the wall temperature, while `nexch` defines the frequency of update to the remaining surface equations — see Section 5.4.10 on page 27 for more info on `nexch`. Default: `50`

h_ablation_0

A rank 1 vector of extent 3 used to compute the heat of ablation in MJ/kg for `quasi_steady` blowing option as

$$\begin{aligned} \texttt{h\_ablation\_0(1)} &+ (\texttt{h\_ablation\_0(2)}) \log p_w \\ &+ (\texttt{h\_ablation\_0(3)})(\log p_w)^2 \end{aligned} \tag{1}$$

where p_w is the local pressure,in atmospheres. Example $\Delta H_{abl} = 28.0 - 1.375 \log p_w + 27.2(\log p_w)^2$. Default: `0.0`

mdot_pressure_0

A rank 1 vector of extent 2 is used to set the blowing or suction distribution defined as

$$\texttt{mdot\_pressure\_0(1)} + (\texttt{mdot\_pressure\_0(2)}) \frac{p}{\rho_\infty V_\infty^2} \tag{2}$$

where p is the local pressure, ρ_∞ is the reference density, and V_∞ is the reference velocity. Positive value produces blowing distribution, while negative value produces suction distribution. Default: `0.0`

`t_sublimation_0`

A rank 1 vector of extent 3 used to compute the sublimation temperature in degrees Kelvin for `quasi_steady` blowing option as

$$\begin{aligned} \texttt{t\_sublimation\_0(1)} &+ (\texttt{t\_sublimation\_0(2)}) \log p_w \\ &+ (\texttt{t\_sublimation\_0(3)})(\log p_w)^2 \end{aligned} \tag{3}$$

where p_w is the local pressure,in atmospheres. Example $T_{sub} = 3797.0 + 342.0 \log p_w + 30.0(\log p_w)^2$. Default: `0.0`

`uncoupled_ablation_flag`

An integer defining if an uncoupled ablation analysis is applied. The uncoupled ablation option is included to provide a baseline solution for the coupled ablation analysis. Default: 0

Options are:

0: Do not apply an uncoupled ablation analysis. It means that the coupled ablation analysis discussed in Section 8.3 on page 70 is applied.

1: Apply an uncoupled ablation analysis to a converged non-ablating flowfield. The procedure for applying the uncoupled ablation analysis is discussed in Section 8.4 on page 72.

`virgin_density_0`

Density of virgin material, kg/m^3. Default: 544.627742, which is for the heritage AVCOAT.

5.4.2 Aerodynamic Coefficient Reference Quantities

`bref`

Yaw moment coefficient reference length, grid-units. Default: `1.0`

`cref`

Pitching moment coefficient reference length, grid-units. Default: `1.0`

`sref`

Reference area for aerodynamic coefficients, grid-units. Default: `1.0`

`xmc`

x-coordinate of moment center, grid-units. Default: `0.0`

ymc

y-coordinate of moment center, grid-units. Default: `0.0`

zmc

z-coordinate of moment center, grid-units. Default: `0.0`

5.4.3 Farfield/Freestream Reference Quantities

alpha

Angle-of-attack in xz plane, degrees, such that

$$u = cos(\alpha)cos(yaw); v = -sin(yaw); w = sin(\alpha)cos(yaw) \qquad (4)$$

where u, v, and w are velocities in x-, y-, and z-coordinate, respectively. Default: `0.0`

density_ref

Free stream density, kg/m^3. Default: `0.001`

rpm

Spin rate, RPM. This is applicable only to axisymmetric cases. Default: `0.0`

tref

Free stream temperature, K. Default: `200.0`

velocity_ref

Free stream velocity, m/s. Default: `5000.0`

yaw

Sideslip angle in xy plane, degrees. Default: `0.0`

5.4.4 Grid Adaptation, Alignment, and Doubling Parameters

beta_grd

This parameter controls grid points normal to the body (k-grid points) are controlled by the following grid stretching function:

$$k_i^* = 1 - \beta \frac{\frac{\beta-1}{\beta+1}^{\frac{k_{max}-k_i}{k_{max}}} - 1}{\frac{\beta-1}{\beta+1}^{\frac{k_{max}-k_i}{k_{max}}} + 1} \qquad (5)$$

where k_i^* are new grid points. The `beta_grd` parameter defines the β coefficient in equation 5. No stretching will be performed if $\beta < 1$. Recommended value is 1.15, if used. Default: `0.0`

ep0_grd

Grid clustering around the shock is designed using the following function:

$$\overline{k^{**}}_i = \epsilon_0 \overline{k_i^*}^2 (1 - \overline{k_i^*})(\overline{k_i^*} + \texttt{fsh}) + \overline{k_i^*} \tag{6}$$

where $\overline{k_i}$ refers to normalized value and k_i^* is defined by equation 5 on the facing page. `ep0_grd` assigns the ϵ_0 coefficient in equation 6. Maximum recommended value is 25/4, if used. Default: `0.0`

kmax_final

The target number of grid cells in the k-direction. Triggered by the global L2 error norm set by `kmax_error`, cells along the k direction are increased by a factor of `kmax_factor` until reaching `kmax_final`. Any value less than the number of k grid cells in `laura.g` file will be ignored, i.e., no coarsening. This option requires all blocks to be active. Default: `0.0`

kmax_error

When the global L2 error norm reaches this value, the number of grid cells in the k direction increases by a factor of `kmax_factor` until the maximum allowable grid cells, `kmax_final`, is reached. This option requires all blocks to be active. Default: `0.01`

fctrjmp

This parameter is used to detect bow shocks. Bow shock is first detected when the sensing parameter defined by `jumpflag` exceeds by this value, while searching from inflow boundary. Default: `1.05`

frac_line_implicit

This positive parameter, which must be less than or equal 1, sets a fraction of the line-implicit direction that is assigned in `assign_tasks` (see Section 5.1 on page 12). This parameter, which supersedes the given assignments in `assign_tasks`, will be applied to all the active blocks. This parameter is recommended where there might be an instability issue such as across strong shocks. Default: `0.7`

fsh

Fraction of arc length distance between body and inflow boundary where bow shock is captured. Default: `0.8`

fstr

This parameter approximately defines fraction of k-cells in boundary layer region. Default: `0.75`

`jumpflag`

An integer flag to select the sensing parameter to be used in detecting position of captured shock for grid adaption. Default: `2`

Options are:

0: Redistributes grid points in k-direction for the target cell Reynolds number defined by `re_cell` parameter, without changing the domain boundary.

1: Selects pressure as the sensing parameter.

2: Selects density as the sensing parameter.

3: Selects temperature as the sensing parameter.

4: Scales the grid distance in the k-direction up by the value defined by `fctrjmp` parameter.

`maxmoves`

An integer number to assign maximum number of times that grid adaption is performed. The value of zero, however, can be used for unlimited number of times. Default: `0`

`max_distance`

This real number defines maximum distance from the body surface that grid outer boundary can be moved away by any one of the flags. This is often useful especially when adapting shock into wake, where the adapting grid may become skewed due to presence of sharp gradients. This value defines the maximum length of the wake. Default: `1.0E+6`

`movegrd`

An integer number for number of cycles between each grid alignment. A zero value disables any grid alignment. Default: `0`

`re_cell`

This value defines the target cell Reynolds number at the wall after a grid movement. Note : If the grid is moving radically Default: `0.1` Note: Use a higher value for `re_cell` for the first grid alignment, if the grid movement causes radical changes in the grid.

The cell Reynolds number is defined as

$$Re_{cell} = \rho c \Delta n / \mu \tag{7}$$

Here ρ is the flow density, c is the sound speed, Δn is the cell height, and μ is the flow viscosity.

5.4.5 Grid File Description

`dimensionality`

A string flag to select the dimensions of the problem. Default: `'3D'`. Available options are:

`'axisymmetric'`

This option selects axisymmetric flow solution. This requires a domain with single-cell width in the j-direction.

`'2D'`

This selects two-dimensional problem, which requires a domain with single-cell width in the j-direction.

`'3D'`

This option solves three-dimensional problems.

`grid_conversion_factor`

This parameter scales the grid to meter. One grid unit equals `grid_conversion_factor` meters. Default: `1`

`single_precision_grid`

Logical flag: `.true.` if grid points in the `laura.g` file are stored as single precision values, otherwise double precision is assumed. Default: `.false.`

5.4.6 Grid Limiter

`g_limiter`

A real number, normally $< 1 \times 10^{-3}$, that limits the cell size of grid cells off the wall for a better quality grid after adaptation. Negative or zero value turns the use of grid limiter off. Default: `0`.

5.4.7 Initialization

`init_vel_fctr`

A real number between `0` and `1` to reduce the initial velocities u, v, w within the domain to avoid creating a vacuum for wake flow problems, which may lead to an invalid solutions. The initial near zero velocity has also shown to ease up the formation of bow shock. Default: `0.01`

5.4.8 Molecular Transport Flags

`ivisc`

An integer flag to engage viscous terms (inviscid=0/viscous=2). Default: `2`

mass_driven_diff

A logical flag to engage binary diffusion driven by mass fraction gradient. Default: .false.

multi_component_diff

A logical flag to engage multi-component diffusion by Stefan-Maxwell equation sub-iterations. Default: .false.

mole_driven_diff

A logical flag to engage binary diffusion driven by mole fraction gradient. Default: .true.

navier_stokes

A logical flag to select equation set for Thin-Layer Navier Stokes or Full Navier Stokes. The navier_stokes = .false. may be used to select Thin-Layer Navier Stokes. Default: .false.

pressure_diffusion

A logical flag to engage pressure diffusion term on Stefan-Maxwell approximation. Default: .false.

schmidt_number

A constant Schmidt number may be specified to calculate diffusivities. If the value is negative, diffusivities are computed directly from collision cross sections. Default: -1.0

prandtl_number

A constant Prandtl number may be specified to calculate conductivities. If the value is negative, conductivities are computed directly from collision cross sections. Default: -1.0

5.4.9 Output Parameters

blayer_io_freq

Number of cycle between save of laura_blayer.dat file – see Section 6 on page 55. Negative value makes the write-out frequency to be the same as iterwrt. Default: -1

isurf_freq

Number of cycles between saves of laura_surface_dtXXXX.g and laura_surface_dtXXXX.q files, where XXXX will be replaced with an integer number, starting from 1. Time accurate simulation is required to engage this command. Default: -1

`iterwrt`

Number of cycles between saves of all output files – see Section 6 on page 55. Default: `200`

`write_extra_q_variables`

A logical flag to output all the computed variables such as laminar and turbulent viscosity, thermal conductivity, species diffusivity, specific heat, enthalpy, species pressure Jacobians, turbulent into `laura.q` file. Default: `.false.`

5.4.10 Numerical Parameters

`augment_shock_dissipation`

Augment dissipation across shock to make cell Re number approach 2 where pressure ratio is greater than 3. Default: `.true.`

`cfl1`

Initial value of CFL number. Default: `5.0`

`cfl2`

Final value of CFL number. Default: `5.0`

`epsa`

Eigenvalue limiting factor. Default: `0.3`

`hrs`

The maximum total CPU time for simulation, hours. Default: `10`

`iramp`

Number of cycles to ramp from `cfl1` to `cfl2`. Default: `200`

`irest`

An integer flag to start the simulation either using freestream values (`irest = 0`), or using the existing solution from `laura.rst` file (`irest = 1`). Default `0`

`jupdate`

Number of cycles between jacobian updates. Default: `10`

`ncyc`

Number of global iterations. Default: `1000`

`nexch`

Number of cycles between exchange of data between processors and updating boundary conditions. Default: `2`

nitfo

Number of cycles using 1st-order spatial accuracy. Default: `0`

nordbc

Boundary condition calculation using 1^{st}-order (`nordbc = 1`) or 2^{nd}-order (`nordbc = 2`) accuracy. Default: `1`

ntran

Number of cycles between transport properties update. Default `1`

rf_inv

Inviscid relaxation factor. Default: `3.0`

rf_vis

Viscous relaxation factor. Default: `1.0`

rf_chem

Chemical source term reduction factor sometimes useful to "ease in" simulations very close to equilibrium. This factor, which must be greater than 1.0 when it is used, changes the answers and must ultimately equal 1.0 in the final simulation. Default: `1.0`

rmstol

A real number to stop the iterations after L2 norm residual reaches this value. Default: `1.0E-10`

sub_iterations

An integer number to control multiple passes through the linear solver, which has been found to increase robustness for some high energy applications. Default: `1`

5.4.11 Radiation Flags

radiation

A logical flag to enable coupling between radiation equation(s) and flow equations. See Section 8.2 on page 69 for coupled radiation procedure. Default: `.false.`

radiation_input_only

A logical flag to enable creation of input file `hara_out.m` to compute radiation outside of LAURA when `radiation` is `.false.` Default: `.false.`

maxrad

An integer number to assign maximum number of calls to radiation interface. The value of zero can be used for unlimited number of times. Default: 0

nrad

Number of iterations between calls to HARA. Default: 3000

iinc_rad, jinc_rad

Increment between i and j lines, respectively at which radiation profile is computed. Interim lines are interpolated. Default: 3

frac_rad_new

Relaxation factor on radiation. $\nabla(q_{rad}) = frac\_rad\_new\nabla(q_{rad})^n + (1 - frac\_rad\_new)\nabla(q_{rad})^{n-1}$. Default: 1.0

absorptivity

Fraction of incident radiative energy absorbed by the wall. Affects surface energy balance and computation of radiative equilibrium wall temperature. Default: 1.0

tw_rad_flag

An integer flag to engage Tauber-Wakefield approximation for radiation cooling on surface-energy balance (on=1/off=0). Default: 0

5.4.12 Solid Surface Boundary Condition Flags

catalysis_model_0

A character identifier for selecting catalysis model for surface type 0 (see Section 5.3 on page 14). This flag is good only for multi-species reacting gases and will be ignored for single-specie gas models. The catalysis model name must be surrounded by quotation marks, e.g., ' '. Default: 'super-catalytic'

Available catalysis models are:

'CCAT-ACC'

This option uses the following surface catalytic coefficients [3] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = \begin{cases} 13.5e^{-8350/T_w^*} & T_w^* \leq 1359.0\ K \\ 5.0 \times 10^{-8} e^{18023/T_w^*} & T_w^* > 1359.0\ K \end{cases} \tag{8}$$

$$\gamma_N = \begin{cases} 4.0e^{-7625/T_w^*} & T_w^* \leq 1475.0\ K \\ 6.2 \times 10^{-6} e^{12100/T_w^*} & T_w^* > 1475.0\ K \end{cases} \tag{9}$$

where T_w^* is calculated as

$$T_w^* = \begin{cases} \min(\ \max(1255.0, T_w),\ 1659.0) & \text{for } \gamma_O \\ \min(\ \max(1255.0, T_w),\ 1900.0) & \text{for } \gamma_N \end{cases} \tag{10}$$

'CSiC'

This option uses the following surface catalytic coefficients [3] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = 6.415 \times 10^{-4} e^{3498.4/T_w^*} \tag{11}$$

$$\gamma_N = 3.993 \times 10^{-4} e^{4402/T_w^*} \tag{12}$$

where T_w^* is given as

$$T_w^* = \min(\ \max(1100.0, T_w),\ 1920.0) \tag{13}$$

'CSiC-SNECMA'

This option uses the following surface catalytic coefficients [3] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = \gamma_N = 9.593 \times 10^{-5} e^{7002.9/T_w^*} \tag{14}$$

where T_w^* is given as

$$T_w^* = \min(\ \max(1350.0, T_w),\ 1920.0) \tag{15}$$

'equilibrium-catalytic'

This option sets species concentrations to equilibrium values at wall pressure and temperature based on elemental mass fractions in the cell above the solid surface boundary.

'fully-catalytic'

This option assumes that all the atomic and ionized oxygen, nitrogen, carbon, and so forth catalyzes to molecular oxygen, nitrogen, carbon, and so on, respectively; i.e.,

$$\gamma_O = \gamma_N = \gamma_C = ... = 1 \tag{16}$$

'non-catalytic'

This option assumes that no atomic or ionized oxygen, nitrogen, carbon, and so forth catalyzes to molecular oxygen, nitrogen, carbon, and so on, respectively; i.e.,

$$\gamma_O = \gamma_N = \gamma_C = ... = 0 \tag{17}$$

'RCC-LVP'

This option uses the following surface catalytic coefficients [3] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = \begin{cases} 7.5e^{-8283/T_w^*} & T_w^* \leq 1499.0\ K \\ 2.5 \times 10^{-7} e^{17533/T_w^*} & T_w^* > 1499.0\ K \end{cases} \tag{18}$$

$$\gamma_N = \begin{cases} 6.0 \times 10^{-2} e^{-2605/T_w^*} & T_w^* \leq 1529.0\ K \\ 1.5 \times 10^{-5} e^{10080/T_w^*} & T_w^* > 1529.0\ K \end{cases} \tag{19}$$

where T_w^* is calculated as

$$T_w^* = \begin{cases} \min(\ \max(1255.0, T_w),\ 1799.0) & \text{for } \gamma_O \\ \min(\ \max(1255.0, T_w),\ 1954.0) & \text{for } \gamma_N \end{cases} \tag{20}$$

`'Scott-RCG'`

This option uses the following surface catalytic coefficients [4] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = 16.0 e^{-10271/T_w} \quad 1400 \leq T_w \leq 1650 \tag{21}$$

$$\gamma_N = 7.14 \times 10^{-2} e^{-2219.0/T_w} \quad 1090 \leq T_w \leq 1670 \tag{22}$$

The same equations will be used even if the wall temperature, T_w, is out of the specified range, in which case a warning will be issued to the `stdout`.

`'Stewart-RCG'`

This option uses the following surface catalytic coefficients [3] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = \begin{cases} 5.0 \times 10^{-3} e^{-400/T_w} & T_w \leq 502 \\ 1.6 \times 10^{-4} e^{1326/T_w} & 502 < T_w \leq 978 \\ 5.2 e^{-8835/T_w} & 978 < T_w \leq 1617 \\ 39 \times 10^{-9} e^{21410/T_w} & 1617 < T_w \end{cases} \tag{23}$$

$$\gamma_N = \begin{cases} 5.0 \times 10^{-4} & T_w \leq 465 \\ 2.0 \times 10^{-5} e^{1500/T_w} & 465 < T_w \leq 905 \\ 10.0 e^{-10360/T_w} & 905 < T_w \leq 1575 \\ 6.2 \times 10^{-6} e^{12100/T_w} & 1575 < T_w \end{cases} \tag{24}$$

`'super-catalytic'`

This option sets the species mass fractions to free stream values as defined in `tdata`—see Section 5.5 on page 36.

`'Zoby-RCG'`

This option uses the following surface catalytic coefficients [5] for catalyzing atomic oxygen and nitrogen to molecular oxygen and nitrogen, respectively.

$$\gamma_O = 9.41 \times 10^{-3} e^{-658.9/T_w} \quad 900 \leq T_w \leq 1500 \tag{25}$$

$$\gamma_N = 7.14 \times 10^{-2} e^{-2219.0/T_w} \quad 1090 \leq T_w \leq 1670 \tag{26}$$

The same equations will be used even if the wall temperature, T_w, is out of the specified range, in which case a warning will be issued to the screen.

emiss_a_0, emiss_b_0, emiss_c_0, emiss_d_0

Real number values to calculate emissivity, ϵ, for solid surface boundary type (see Section 5.3 on page 14) from the following equation:

$$\epsilon = \epsilon_a + \epsilon_b T_w + \epsilon_c T_w^2 + \epsilon_d T_w^3 \tag{27}$$

where T_w is the surface temperature. Values for ϵ_a, ϵ_b, ϵ_c, and ϵ_d are defined by emiss_a_0, emiss_b_0, emiss_c_0, and emiss_d_0, respectively. Default: emiss_a_0=0.89, emiss_b_0=0.0, emiss_c_0=0.0, emiss_d_0=0.0

ept

Under-relaxation parameter for radiative equilibrium wall temperature:

$$q_{i,j}^{n+1} = (1 - \texttt{ept} q_{i,j}^{n}) + \texttt{ept} q_{i,j}^{n+1} \tag{28}$$

where n denotes iteration level, $q_{i,j}^{n+1}$ is the most recent value of convective heat flux, $q_{i,j}$, $q_{i,j}^{n}$ is the value of convective heat flux from the previous iteration as adjusted by previous application of this formula, and $q_{i,j}^{n+1}$ is the newly adjusted value for convective heat flux at the $n+1$ iteration level. Default: 0.01

surface_temperature_type_0

Character identifier for surface temperature model selection. Default: 'constant'

Options are:

'adiabatic'

Surface temperature will be such that there is no heat transfer between the surface and the gas adjacent to the surface.

'constant'

The surface temperature stays constant as given by twall_bc value.

'radiative equilibrium'

The surface temperature is calculated so that the heat flux to the wall, q_w, is in equilibrium with radiation heat flux:

$$q_w = \epsilon \sigma T_w^4 \tag{29}$$

where σ is the Stefan-Boltzmann constant, and ϵ is the surface emissivity defined by emiss_a_0, emiss_b_0, emiss_c_0, emiss_d_0 values.

`'surface energy balance'`

This option is required for ablating surfaces to compute surface temperature as a function of the surface energy balance and the relevant surface chemistry kinetics.

`surface_group_name_0`

Character descriptor for surfaces with solid surface boundary types (see Section 5.3 on page 14). Any character can be specified to group solid surface boundaries. Default: `'default surface 0'`

`twall_bc`

Initial wall temperature for solid surface boundaries, K. (See Section 5.3 on page 14.) The wall temperature stays constant as specified by this parameter if `surface_temperature_type_0 = 'constant'`. Default: `500.0`

5.4.13 Surface Recession Flags

`shape_change`

A logical flag engaging the geometry shape change due to ablation. Default : `.false`. A trajectory file is required if `shape_change=.true`. — see Section 5.11 on page 47.

5.4.14 Thermochemical Nonequilibrium Flags

`augment_kinetics_limiting`

A logical flag engaging extra limiting on reaction rates by modifying upper and lower temperature limits in forward and backward rates. Upper and lower limits on rate constants may be specified for individual reactions in the file `kinetic_data`. Default: `.false`.

`chem_flag`

An integer flag to engage chemical source term for non-equilibrium flow (on=1/off=0). Default: `0`

`therm_flag`

An integer flag to engage thermal source term for non-equilibrium flow (on=1/off=0). Default: `0`

5.4.15 Time Accurate Flags

`itime`

An integer flag to engage time accuracy: `itime = 0` produces 0th-order in time, `itime = 1` produces 1st-order in time, `itime = 2` produces 2nd-order in time. This flag is superseded by value of `time_step`. A positive

value for time_step forces temporal accuracy of at least first-order. A negative or unspecified value for time_step forces itime=0. Default: 0

subiters

An integer number to specify number of iterations between each time step for time-accurate simulations. Default: 10

time_step

A positive real value to specify nondimensional time step for time accurate simulation. A negative or zero value supersedes itime value and disables time accurate simulation. Default: -1

5.4.16 Trajectory Related Flags

trajectory_data_point

Pickup simulation from this line in the file laura_trajectory_data.

Default: 0 if laura_trajectory_data is not present.

Default: 1 if laura_trajectory_data is present.

Note that the reference quantities are defined consistently across the trajectory using the values supplied in namelist above. The reference quantities are NOT reset according to the time varying free stream conditions. Consequently, dimensionless values of density and velocity in free stream will not generally equal 1. Dimensionless values of total enthalpy in free stream will not equal 0.5 which may impact post-processing tools that are key on a specific number for total enthalpy to detect the boundary-layer edge.

The algorithm is most efficient if the restart solution is converged (or nearly converged so that line-implicit relaxation may be applied) at free stream conditions equal to the reference quantities in laura_namelist_data. If one wishes to pickup a computation for trajectory_data_point > 1 then it is best to start at the converged restart file for the previous trajectory point. Restart files and post processing files for each trajectory point have the trajectory point number included as part of the root name. Thus if one wants to pickup at trajectory point 12 one should copy laura_0011.rst to laura.rst and (if grid is being updated) laura_0011.g to laura.g. This advice is offered because restart solutions are converted according to the ratio of density and velocity between adjacent trajectory points to bring interior initial conditions closer to the new inflow boundary conditions.

5.4.17 Turbulent Transport Models

`coupled_tke`

Logical flag indicating if turbulent kinetic energy is treated as part of the total energy in the energy equation. Default:.`true`.

`prandtl_turb`

Turbulent Prandtl number. Default: `0.9`

`schmidt_turb`

Turbulent Schmidt number. Default: `0.9`

`turb_int_inf`

Farfield turbulent kinetic energy non-dimensionalized by square of the reference velocity. If negative, use recommended far field boundary conditions: $(\rho k)_\infty = 0.01\mu_\infty$ and $(\rho\omega)_\infty = 2$. Default: `-0.0001`

`turb_model_type`

An integer flag to select turbulence modeling. Default: `0`

Options are:

`-1`: Baldwin-Lomax algebraic turbulence model. [6]

`-2`: Cebeci-Smith algebraic turbulence model. [6]

`0`: Laminar flow, i.e., no turbulence model.

`1`: Spalart Almaras one-equation turbulence model.

`2`: Menter two-equation turbulence model.

`3`: Menter-SST two-equation turbulence model.

`4`: Wilcox k-ω 1998 turbulence model.

`6`: Wilcox k-ω 2006 turbulence model.

`turb_comp_model`

Integer flag (1=on, 0=off) to engage compressibility correction for `turb_model_type` = 2, 3, 4. Sometimes observed to cause spike in turbulent kinetic energy crossing bow shock. Expect small effect in boundary layer but large effect in free mixing layer. Default: `0`

`turb_vis_ratio_inf`

Farfield ratio of turbulent to laminar viscosity. Default: `0.001`

`xtr`

Transition location along x-axis in grid units. Engaged only for algebraic models. Default: `0.0`

5.4.18 Venting Boundary Condition Flags

vacuum_pressure_coefficient

This parameter, which is in N/m^2 sets the pressure coefficient behind type 6 boundaries (see Section 5.3 on page 14), which forces flow out of the domain. This coefficient is defined as

$$\texttt{vacuum\_pressure\_coefficient} = \frac{p_0 - p_\infty}{2}. \tag{30}$$

where p_0 is back pressure where the gas is venting out, and p_∞ is farfield pressure. Default: `0.0`

vacuum_pressure_factor

Factor on pressure across type 6 boundary to force flow out of domain.

$$\texttt{vacuum\_pressure\_factor} = \frac{p_0}{p_1}, \tag{31}$$

where p_1 is the pressure just before the gas vents out. Default: `0.01`

5.5 tdata

The gas model is defined in this file. It contains a list of key words, sometimes followed by numeric values, which identify components of the gas model. One or more spaces must be present between keyword and values when appearing on the same line. Spaces may appear to the left or right of any key word. The first line of the file must not be blank, however.

The following subsections describe available gas model options.

5.5.1 Perfect Gas

The perfect-gas option is engaged with either of the following keywords: `perfect_gas`, `PERFECT_GAS`, `Perfect_Gas`, or `Perfect_gas`.

If no further data is provided in this file, this single line `tdata` file will assume the following parameter values in SI units:

```
gamma   =   1.4
mol_wt  =  28.8
suther1 =   0.1458205E-05
suther2 = 110.333333
prand   =   0.72
```

Here, `gamma` is the gas specific heat ratio, `mol_wt` is the gas molecular weight, `prand` is the gas Prandtl number, and `suther1` and `suther2` are the first and second Sutherland's viscosity coefficients, s_1 and s_2, respectively, defined as

$$\mu = s_1 \frac{T^{3/2}}{T + s_2} \tag{32}$$

These values can be modified and explicitly defined in the `tdata` by the keyword `&species_properties` in the second line followed by the gas parameters and `/` at the last line of the file. For example,

```
perfect_gas
&species_properties
gamma = 1.4
mol_wt = 28.0
suther1 = 0.1E-05
suther2 = 110.3
prand   = 0.7
/
```

5.5.2 Equilibrium Gas

To engage the Tannehill curve fits for thermodynamic and transport properties of equilibrium air [7], the following keyword should be used in the first line of the `tdata` file: `equilibrium_air_t`. No additional inputs or files are required to engage the Tannehill option for equilibrium air.

To use a table look-up capability for equilibrium gases [8], the following keyword should be placed in the `tdata` file, instead: `equilibrium_air_r`. Note that this option still uses the Tannehill transport properties.

5.5.3 Mixture of Thermally Perfect Gases

If the gas is a mixture of thermally perfect gases and multi-species transport solution is desired the species names followed by their mass fractions must be provided in the `tdata` file. Thermal state of the gas may be defined as the first entry by either of the following flags:

`one`

This flag assumes that all the species are thermally in equilibrium state. That is translational temperature, T, and vibrational temperature, T_v are equal. This is known as *one-temperature*, 1-T, model.

`two`

This flag assumes that energy distribution in the translational and rotational modes of heavy particles (not electrons) are equilibrated at translational temperature, T, and all other energy modes (vibrational, electronic, electron translational) are equilibrated at vibrational temperature, T_v. This is known as *two-temperature*, 2-T, model.

`FEM`

This option, called Free-Energy Minimization, causes the species continuity equations to be replaced with elemental continuity equations and equilibrium relations for remaining species.

One temperature model is assumed if thermal state of the gas is not provided in the first line of the `tdata` file. In this case, the first line must contain species information. Note, the first line must not be blank.

Subsequent file entries include species names and their mass fractions at freestream/farfield boundary. Only one specie per line is allowed. The species mass fraction at the boundary is defined in the same line as the species name separated by one or more spaces. If no value appears to the right of the species name then that species is assumed not to be present at the boundary but may be produced through chemical reactions elsewhere in the flowfield.

Example 1: 1-T, 5-species air model: In this example, only molecular oxygen and nitrogen are present on freestream/farfield boundary, but atomic nitrogen and oxygen and nitric oxide may be produced elsewhere in the flow field due to chemical reactions.

```
one
N2  .767
N
O2  .233
O
NO
```

Example 2: 2-T, 11-species air model: In this example, the gas is assumed to be a mixture of 11 thermally perfect gases. A solution to a thermal non-equilibrium state of the gas is also desired (2-T model).

```
two
N2  .767
N
O2  .233
O
NO
O2+
O+
NO+
e-
```

5.6 hara_namelist_data

This file controls the radiation models used by the HARA radiation module [9, 10]. It is optional for coupled radiation simulations. If it is not present, then the code automatically chooses the radiative mechanisms associated with species present in the flowfield (and have number densities greater than 1000 particles/cm^2), and other options are set to the defaults listed in the following

section. For users not experienced in shock-layer radiation, it is recommended that this `hara_namelist_data` file not be applied (meaning it is removed from the working directory), therefore allowing the radiative mechanisms to be automatically chosen and the default model options applied.

5.6.1 Specifying radiation mechanisms for atomic species

The treatment of radiation resulting from atomic lines, atomic bound-free and free-free photoionization (referred to here as atomic continuum), and negative ion continuum is available for atomic carbon, hydrogen, oxygen, and nitrogen. These mechanisms are specified through the following binary flags (on=1/off=0). If any of these flags are not present in `hara_namelist_data`, then that flag is set to true only if the number density of the associated atomic specie is greater than 1000 particles/cm^2 somewhere in the flowfield.

`treat_[?]_lines`

A binary flag to enable the treatment of atomic lines for specie `[?]`, where `[?]` can be c, h, n, and o, for atomic carbon, hydrogen, nitrogen and oxygen, respectively.

`treat_[?]_cont`

A binary flag to enable the treatment of atomic bound-free and free-free continuum for specie `[?]`, where `[?]` can be c, h, n, and o, for atomic carbon, hydrogen, nitrogen and oxygen, respectively.

`treat_[?]_other`

A binary flag to enable the treatment of the negative-ion continuum for specie `[?]`, where `[?]` can be c, h, n, and o, for atomic carbon, hydrogen, nitrogen and oxygen, respectively.

5.6.2 Specifying radiation mechanisms for molecular species

The treatment of radiation resulting from numerous molecular band systems is available through the following binary flags (on=1/off=0). If any of these flags are not present in `hara_namelist_data`, then that flag is set to true only if the number density of the associated molecular specie is greater than 1000 particles/cm^2 somewhere in the flowfield. Additional band systems are listed in Appendix B on page 78. These additional band systems are generally considered negligible relative to those listed in this section, and therefore for computational efficiency, they are not engaged by default. Definitions of each band system and the modeling data applied are discussed in Refs. [9, 11].

`treat_band_c2_swan`

A binary flag activating the C_2 Swan band system.

treat_band_c2h

A binary flag activating the C_2H band system.

treat_band_c3

A binary flag activating the C_3 and Vacuum Ultra-Violet (VUV) band systems.

treat_band_cn_red

A binary flag activating the CN red band system.

treat_band_cn_violet

A binary flag activating the CN violet band system.

treat_band_co4p

A binary flag activating the CO 4+ band system.

treat_band_co_bx

A binary flag activating the CO B-X band system.

treat_band_co_cx

A binary flag activating the CO C-X band system.

treat_band_co_ex

A binary flag activating the CO E-X band system.

treat_band_co_ir

A binary flag activating the CO X-X band system.

treat_band_h2_lyman

A binary flag activating the H_2 Lyman band system.

treat_band_h2_werner

A binary flag activating the H_2 Werner band system.

treat_band_n2fp

A binary flag activating the N_2 1+ band system.

treat_band_n2sp

A binary flag activating the N_2 2+ band system.

treat_band_n2pfn

A binary flag activating the N_2^+ first-negative band system.

treat_band_n2_bh1

A binary flag activating the N_2 Birge-Hopfield I band system.

`treat_band_n2_bh2`

A binary flag activating the N_2 Birge-Hopfield II band system.

`treat_band_no_beta`

A binary flag activating the NO beta band system.

`treat_band_no_delta`

A binary flag activating the NO delta band system.

`treat_band_no_epsilon`

A binary flag activating the NO epsilon band system.

5.6.3 Atomic line models

There are various models available for atomic line radiation, one of which must be chosen for each specie that engages atomic line radiation (as specified using `treat_[?]_lines`). This choice of atomic line model is made using the following flags. The listed defaults are applied if the individual flag is not present in `hara_namelist_data`, or if `hara_namelist_data` is not present in the working directory. All model types in this category must be surrounded by a quotation marks, e.g. ‘ ’.

`c_atomic_line_model`, `h_atomic_line_model`

A character identifier for selecting the atomic line model for atomic carbon or hydrogen. Presently, the only available option is the model compiled in Ref. [11], which is referred to here as the Complete Line Model (CLM). Default : `‘clm’`

`n_atomic_line_model`, `o_atomic_line_model`

A character identifier for selecting the atomic line model for atomic nitrogen or oxygen. The available models are compiled and compared in Ref. [9], which is referred to here as the Complete Line Model (CLM). Default : `‘clm’` Available models are:

`‘all multiplets’`

This model treats all lines as grouped multiplets. This significantly reduces the number of lines treated as well as the computational expense. However, this grouped multiplet approximation will lead to errors for non-optically-thin conditions.

`‘clm’`

This model, which stands for Complete Line Model, applies the individual treatment of strong atomic lines while applying multiplet averages for weak lines. This is the recommended model.

5.6.4 Electronic state population models

These flags specify the model applied for predicting the electronic state populations of atoms and molecules. The listed defaults are applied if the individual flag is not present in `hara_namelist_data`, or if `hara_namelist_data` is not present in the working directory. All model types in this category must be surrounded by a quotation marks, e.g. ‘ ’.

Atomic electronic states

The electronic state populations for atoms are required for computing atomic line and photoionization emission and absorption. The compilation and comparison of the available models are presented in Ref. [10].

`c_electronic_state`, `h_electronic_state`

A character identifier for selecting the electronic state model for atomic carbon and hydrogen. Available models are (Default : ‘boltzmann’):

‘`boltzmann`’

Applies Boltzmann population of electronic states.

‘`Gally_1st_order_LTNE`’

Applies the Gally first-order local thermodynamic nonequilibrium method [12], which approximately accounts for the non-Boltzmann population of atomic states.

`n_electronic_state`, `o_electronic_state`

A character identifier for selecting the electronic state model for atomic nitrogen and oxygen. Available models are (Default : ‘CR’):

‘`boltzmann`’

Same as for `c_electronic_state`

‘`Gally_1st_order_LTNE`’

Same as for `c_electronic_state`

‘`CR`’

Applies the detailed Collisional Radiative (CR) model developed in Ref. [10].

‘`AARC`’

Applies the Approximate Atomic Collisional Radiative (AARC) model developed in Ref. [10]. This model is essentially a curve-fit based approximation of the CR model, which allows for improved computational efficiency with a slight loss in accuracy.

Molecular electronic states

The electronic state populations for molecules are required for computing molecular band emission and absorption. The compilation and comparison of the available models are presented in Refs. [10, 13].

`molecular_electronic_state`

A character identifier for selecting molecular electronic state for all molecular band systems. Available models are (Default : 'CR'):

`'boltzmann'`

Applies Boltzmann population of electronic states.

`'CR'`

Applies a detailed Collisional Radiative model considering both heavy-particle and electron impact transitions. Some molecular states are still assumed Boltzmann with this model because no data is presently available for the CR model.

5.6.5 Other flags

`use_triangles`

A logical flag specifying whether optically-thin atomic lines are modeled as triangles to reduce computational time. This option has shown to result in a negligible loss of accuracy while greatly reducing the computational time, [9] and is therefore recommended. Default : .true. Note: This flag is automatically set to `.true.` when `n_` or `o_atomic_line_model= 'clm'` — see Section 5.6.3 on page 41.

`use_edge_shift`

A logical flag to engage the photoionization edge shift [9] for atomic bound-free radiation. (on=1/off=0). Default : .true.

5.7 jdata

This file defines jets chamber conditions at faces of blocks with characteristic boundary condition [14]. Jets species names and their mass fractions are also specified in this file. There must be one entry for each jet. Each jet can have a different chamber condition and species composition.

5.7.1 Multi-species Jet Composition

Consider a system with two jets; i.e., there are two characteristic boundary condition entries in the `laura_bound_data` file. N_2 and H_2 are being fired from one of the jets while pure H_2 is being used for the other jet. Note: The sum

of the species mass fraction must equal 1.0 or the code stops with an error message. In this example, the `jdata` file should look like the following:

```
&jet_properties                                                    1
nozzle_grid_block     = 9                                          2
plenum_t0             = 401.7                                      3
plenum_p0             = 2434000.                                   4
jet_number_of_species = 2                                          5
/                                                                  6
N2 0.8                                                             7
H2 0.2                                                             8
                                                                   9
&jet_properties                                                    10
nozzle_grid_block     = 6                                          11
plenum_t0             = 105.7                                      12
plenum_p0             = 10250.                                     13
/                                                                  14
H2 1.0                                                             15
```

It should be noted here that the `tdata` must contain all the jet effluent species. The default value for `jet_number_of_species` is `1`. Jet properties can be placed in this file in a random order.

5.7.2 Perfect Gas Jet Composition

If `perfect_gas` is specified in the first line of the `tdata` file, the code ignores jet species composition even they are left in the `jdata` and assumes perfect gas with the same molecular weight and Prandtl number as specified in the `tdata` file.

```
&jet_properties                                                    1
nozzle_grid_block     = 19                                         2
plenum_t0             = 401.7                                      3
plenum_p0             = 2434000.                                   4
jet_number_of_species = 1                                          5
/                                                                  6
N2 0.8                                                             7
H2 0.2                                                             8
                                                                   9
&jet_properties                                                    10
nozzle_grid_block     = 36                                         11
plenum_t0             = 105.7                                      12
plenum_p0             = 10250.                                     13
/                                                                  14
H2 1.0                                                             15
                                                                   16
```

```
...                                                                    17
...                                                                    18
...                                                                    19
```

5.8 kinetic_data

This file defines possible chemical reactions and is optional. By default, `kinetic_data` is read from `[install-prefix]/share/physics_modules`, but if present in the local run directory, LAURA will read reactions from the local file instead.[14]

Reactants and products can be any species defined in the `species_thermo_data` file—see Section 5.13 on page 49. A sample entry looks like this,

```
O2 + M    <=>   2O + M                                                 1
2.000e+21  -1.50  5.936e+04                                           2
teff1 = 2                                                              3
exp1 = 0.7                                                             4
t_eff_min = 1000.                                                      5
t_eff_max = 50000.                                                     6
C = 5.0                                                                7
O = 5.0                                                                8
N = 5.0                                                                9
H = 5.0                                                                10
Si = 5.0                                                               11
e- = 0.                                                                12
```

The first line specifies the reaction while line 2 provides three coefficients of an Arrhenius-like equation,

$$K_f = c_f T_{eff}^{\eta} e^{-\epsilon_0/kT_{eff}} \tag{33}$$

where c_f is the pre-exponential factor, η is the power of temperature dependence on the pre-exponential factor, ϵ_0 is the Arrhenius activation energy, and k is the Boltzmann constant. The arrowheads in line 1 signify the allowed directionality of the reaction. The symbol => denotes forward reaction only while <=> denotes forward and backward rates are computed. The coefficients in line 2 correspond to c_f, η, and ϵ_0/k, respectively. For reactions with a generic collision partner, `M`, such as this one, these coefficients correspond to Argon; and other collision partners and their efficiencies (multipliers of c_f) are specified on lines following line 5 and 6, which give the valid temperature range for the reaction. The effective temperature, T_{eff}, is defined according to a given integer number in line 3; Default: 2

[14] The precise installation location is given by LAURA during startup. It can also be found on Unix-like systems from the executable itself by issuing the command `strings laura | grep share`.

teff1,teff2

Flag defining formula to compute the effective temperature T_{eff} for the forward rate and backward rate, respectively. It is engaged for the case of thermal nonequilibrium. Options for `teff` are:

1: $T_{eff} = T_{tr}$

2: $T_{eff} = T_{tr}^{exp1} T_v^{1-exp1}$

3: $T_{eff} = T_v$

where T_{tr} and T_v are translational and vibrational temperatures, respectively. Default: `1`

exp1

The exponent used to define the effective temperature when `teff1` = 2 (forward rate) or `teff2` = 2 (backward rate). See previous equations for `teff` options. Default: `0.7`

rf_max

Upper limit on forward reaction rate in cgs units engaged only if `augment_kinetics_limiting` is true. See output file `kinetic_diagnostics_output` for unlimited rates as function of temperature. Default: `1.e+20`

rf_min

Lower limit on forward reaction rate in cgs units engaged only if `augment_kinetics_limiting` is true. See output file `kinetic_diagnostics_output` for unlimited rates as function of temperature. Default: `1.e-30`

rb_max

Upper limit on backward reaction rate in cgs units engaged only if `augment_kinetics_limiting` is true. See output file `kinetic_diagnostics_output` for unlimited rates as function of temperature. Default: `1.e+30`

rb_min

Lower limit on backward reaction rate in cgs units engaged only if `augment_kinetics_limiting` is true. See output file `kinetic_diagnostics_output` for unlimited rates as function of temperature. Default: `1.e-30`

t_eff_min

The minimum temperature for T_{eff} to compute reaction rates to circumvent stiff source terms. Default: `1000.`

t_eff_max

The maximum temperature for T_{eff} to compute reaction rates to circumvent stiff source terms. Default: `50000.`

5.9 laura.rst

This Fortran-unformatted binary file has the flowfield solution for every grid cell, boundary surface values, and grid cell derivatives for each block face. The number of variables in this file varies depending on number of conservation equations that LAURA needs to solve as specified through the `tdata` and `laura_namelist_data` files. A file of this format, but named `laura_new.rst`,[15] is generated by LAURA at the end of a successful run. The `laura.rst` file is required only if restarting from a prior run, i.e., `irest = 1` in `laura_namelist_data` as described in Section 5.4.10 on page 27.

5.10 laura.trn

Turbulent transition location and transition length are specified in `laura.trn`.[16] One line of data consisting of four integers followed by at least one space with the following entries per block is required: `block_number`, `turbulent_flag`, `transition_location`, and `transition_length_factor` where

`turbulent_flag`

One of `-1`, `0`, or `1`, which correspond to laminar, transition location in block, or fully turbulent flow, respectively.

`transition_location`

Specified as an x coordinate. This value used only if `turbulent_flag` is `0`.

`transition_length_factor`

Defined as (transition length)/L, where L is the distance from the nose to the transition location. Use `0` for instantaneous transition.

5.11 laura_trajectory_data

This file is used to sequentially simulate points along a trajectory. Each line of the file defines a single trajectory point. The trajectory point data is entered in free format with time, velocity, density, temperature, alpha, and yaw in MKS units (6 entries per line). The simulation will start at trajectory point 1 by default unless another line number is specified in the `laura_namelist_data` using the namelist variable `trajectory_data_point` — see Section 5.4.16 on page 34.

The restart solution files and post processing files for each trajectory point are saved with a four digit number added to the usual root name of these files corresponding to the trajectory point number.

[15] When running a trajectory sequence, the file will be named `laura_####.g` where `####` is the trajectory point index.

[16] LAURA transition files from versions prior to 5 can also be used.

5.12 laura_vis_data

This file contains the data set that overrides the default viscous terms in the i-, j-, and k-directions. There are four integers consisting of block number and toggles for each viscous term per line in this file: `blk_num`, `ivis`, `jvis`, and `kvis` where `ivis`, `jvis`, and `kvis` are viscous terms in the i-, j-, and k-directions, respectively. The viscous-term flag options are:

`0`: Viscous terms are not engaged in the respective direction; i.e. inviscid flow.

`1`: Viscous terms and a reduced eigenvalue limiter are engaged in the respective direction. The reduced eigenvalue limiter is to prevent distortion of computed heating.

`2`: Viscous terms are engaged but an unmodified eigenvalue limiter is retained to maintain stability.

The following defaults will be applied if this file is not present in the working directory:

If `ivisc = 0` in `laura_namelist_data` (see Section 5.4 on page 17) then inviscid flow is specified and `ivis`, `jvis`, and `kvis` are set to zero for all blocks by default.

If `ivisc = 2` and `navier_stokes = .false.` in the `laura_namelist_data` file then the default is a function of the wall boundary condition. If the wall boundary on the i_{min} or i_{max} face of block n is a solid surface (see Section 5.3 on page 14) then `ivis(n) = 2`, otherwise `ivis(n) = 0`. In like manner, if the wall boundary on the j_{min} or j_{max} face of block n is a solid surface then `jvis(n) = 2`, otherwise `jvis(n) = 0`. The default specification for `kvis(n)` is set to 1 if k_{min} or k_{max} of block n is a solid surface, otherwise `kvis(n) = 0`. This default recognizes that the standard orientation is for the solid wall on the k_{min} surface and the reduced eigenvalue limiter is required in this case.

If `ivisc = 2` and `navier_stokes = .true.` in `laura_namelist_data` then the default is `ivis = 2`, `jvis = 2`, and `kvis = 1` for all the blocks.

There may be circumstances where the user wishes to override these defaults. If a block is away from the stagnation streamline crossing the shock into the boundary layer then a more accurate heating on side walls (i and/or j) will be returned using `ivis` and/or `jvis` set to 1 without sacrificing stability. In the case of cavities, a block may sit over a cavity and not have any solid boundaries itself but has a well defined boundary layer streaming into it from an adjacent block. In this case, even though the block has no solid boundaries itself, it should engage viscous terms to capture the entering shear layer.

Example: To override the default values and to reset viscous terms in block number 3 to `ivis = 1`, `jvis = 0`, and `kvis = 1`, and in block number 5 to `ivis =0`, `jvis =1`, and `kvis =1` the `laura_vis_data` would look like:

```
3 1 0 1
5 0 1 1
```

5.13 species_thermo_data

The `species_thermo_data` file is the master file for species thermodynamic data.[17] Each species record consists of the species name, a species properties namelist - `&species_properties`, the number of thermodynamic property curve fit ranges, and the curve fit coefficients for each range [15]. No blank line is allowed in this file.

Elements of the `&species_properties` namelist are:

`molecule`

A logical flag to determine whether the species is molecule (composed of more than one atom); If the species is molecule then `molecule = .true.`, otherwise `molecule = .false.`

`ion`

A logical flag to identify the charged particle. `ion = .true.` for charged particles except for neutrals and electrons. This flag initializes electron-neutral energy exchange cross section and sum of the charges.

`charge`

An integer number to determine number of positive charges in particle. If `ion = .false.` then `charge = 0`.

`elec_impct_ion`

This real number to set the energy for neutrals (i.e. `ion=.false.`) in electron volts (eV) that is required to liberate an electron when the neutral impacts with free electron.

`mol_wt`

A real number to set the molecular weight of the particle. This parameter is always required.

`siga`

An array of three real numbers, which correspond to curve fit coefficients for electron-neutron energy exchange cross section defined as

$$\sigma_{en} = a + bT + cT^2 \tag{34}$$

where σ_{en} is the electron-neutron energy exchange collision cross section in m^2, a, b, and c are the curve fit coefficients, and T is the gas temperature [16,17]. The format to define these coefficients is `siga=a, b, c`. For example, `siga=7.5e-20, 0, 0`.

[17] The `species_thermo_data` file should only be changed by developers.

disoc_ener

A real number to set dissociation energy of molecule in electron volts (eV).

alantel

A real number to set Landau-Teller constant to compute vibrational relaxation time for molecule. These are defined in Ref. [18]. This variable is required if `molecule=.true.`.

cprt0

A non-dimensional real number that defines translational-rotational heat capacity that is normalized by gas constant. This is equal to

$$cprt() = \frac{f+2}{2} \tag{35}$$

where f is the number of degrees of freedom in translation and rotation. The defaults for atoms and diatomic molecules are 2.5 and 3.5, respectively.

Example: A portion of the `species_thermo_data` that provides thermodynamic properties of carbon is shown below.

```
C                                                                                1
&species_properties                                                              2
molecule = .false.                                                               3
ion = .false.                                                                    4
charge = 0                                                                       5
elec_impct_ion = 11.264                                                          6
siga = 7.5e-20, 5.5e-24, -1.e-28                                                 7
mol_wt = 12.01070                                                                8
/                                                                                9
3                                                                                10
 0.64950315E+03 -0.96490109E+00  0.25046755E+01 -0.12814480E-04                  11
 0.19801337E-07 -0.16061440E-10  0.53144834E-14  0.00000000E+00                  12
 0.85457631E+05  0.47479243E+01   200.000  1000.000                             13
-0.12891365E+06  0.17195286E+03  0.26460444E+01 -0.33530690E-03                  14
 0.17420927E-06 -0.29028178E-10  0.16421824E-14  0.00000000E+00                  15
 0.84105978E+05  0.41300474E+01  1000.000  6000.000                             16
 0.44325280E+09 -0.28860184E+06  0.77371083E+02 -0.97152819E-02                  17
 0.66495953E-06 -0.22300788E-10  0.28993887E-15  0.00000000E+00                  18
 0.23552734E+07 -0.64051232E+03  6000.000 20000.000                              19
```

The species name is defined in line 1. Between lines 2 and 9 species properties are defined. These parameters and/or flags state that the carbon molecular weight is 12.0107, and the species is neither a molecule nor a charged particle, but it can liberate an electron when its energy reaches 11.264 eV after it is impacted with a free electron.

Line 10 shows that there are three thermodynamic property curve fits for temperature ranges of 200 K $< T <$ 1,000 K, 1,000 K $< T <$ 6,000 K, and 6,000 K

$< T <$ 20,000 K. Each data range consists of 12 real numbers with a restriction of 4 real numbers per line. The first 10 real numbers are the thermodynamic curve fit coefficients, and the last two real numbers identify the temperature range for the given curve fit coefficients. These coefficients are used to calculate the following thermodynamic properties

$$c_p(T)/R = a_1T^{-2} + a_2T^{-1} + a_3 + a_4T + a_5T^2 + a_6T^3 + a_7T^4 \tag{36}$$

$$h(T)/RT = -a_1T^{-2} + a_2T^{-1}ln\ T + a_3 + a_4\frac{T}{2} + a_5\frac{T^2}{3} + a_6\frac{T^3}{4} + a_7\frac{T^4}{5} + \frac{a_9}{T} \tag{37}$$

$$s(T)/R = -a_1\frac{T^{-2}}{2} - a_2T^{-1} + a_3ln\ T + a_4T + a_5\frac{T^2}{2} + a_6\frac{T^3}{3} + a_7\frac{T^4}{4} + a_{10} \tag{38}$$

where T is the gas temperature, R is the universal gas constant, c_p, h and s are the species specific heat, enthalpy and entropy, respectively, and a_i are the provided curve fit coefficients in `species_thermo_data`.

The following corrections will be applied if the gas temperature is out of the given range for the given curve fit coefficients:

$$c_p(T) = c_p(T^*) \tag{39}$$

$$h(T) = h(T^*) + (T - T^*)c_p(T^*) \tag{40}$$

$$s(T) = s(T^*) + ln\ \frac{T}{T^*}c_p(T^*) \tag{41}$$

where

$$T^* = \begin{cases} T_{lower} & \text{for } T < T_{lower} \\ T_{upper} & \text{for } T > T_{upper} \end{cases} \tag{42}$$

5.14 `species_transp_data`

This file[18] contains log-linear curve fit coefficients for species collision cross section that are defined based on temperature range of 2,000–4,000 K [19]. This temperature range spans boundary-layer temperatures for typical hypersonic entry. The curve fit for the given coefficients is poor, however, at temperatures below 1,000 K. A higher order curve fit data is available in `species_transp_data_0`, which supersedes that of `species_transp_data`—see Section 5.15 on the following page.

```
 Ar Ar                                                          1
-14.6017 -14.6502 -14.5501 -14.6028 ! trr132+kestin et al       2
Ar+ Ar+                                                         3
-11.48 -12.08 -11.50 -12.10                                     4
Ar N2                                                           5
-14.5995 -14.6475 -14.5480 -14.5981 ! kestin et al              6
Ar CO                                                           7
-14.5975 -14.6455 -14.5459 -14.5964 ! kestin et al              8
```

[18]The `species_transp_data` file should only be changed by developers.

5.15 species_transp_data_0

This file[19] provides collision cross section coefficients for a higher order curve fit data [20, 21] than those are in the species_transp_data file—see Section 5.14. The data in species_transp_data_0 supersedes the data in species_transp_data if the file is placed in the working directory hosting the simulation.

```
O2 N      1 1 1      (c)
         -1.1453028E-03  1.2654140E-02 -2.2435218E-01  7.7201588E+01
         -1.0608832E-03  1.1782595E-02 -2.1246301E-01  8.4561598E+01
          1.4943783E-04 -2.0389247E-03  1.8536165E-02  1.0476552E+00

NO N      1 1 1      (c)
         -1.5770918E-03  1.9578381E-02 -2.7873624E-01  9.9547944E+01
         -1.4719259E-03  1.8446968E-02 -2.6460411E-01  1.0911124E+02
          2.1014557E-04 -3.0420763E-03  2.5736958E-02  1.0359598E+00

NO O      1 1 1      (c)
         -1.0885815E-03  1.1883688E-02 -2.1844909E-01  7.5512560E+01
         -1.0066279E-03  1.1029264E-02 -2.0671266E-01  8.2644384E+01
          1.4145458E-04 -1.9249271E-03  1.7785767E-02  1.0482162E+00

END END   1 1 0
             0. 0. 0. 0.
             0. 0. 0. 0.
```

5.16 surface_property_data

This file is required if there are more than 1 solid surface boundary types (see Section 5.3 on page 14) and the surface conditions of these solid surfaces differ from those specified in laura_namelist_data (see Section 5.4 on page 17).

Surface boundary properties of each solid surfaces must be bounded by:

```
&surface_properties

/
```

The first instance of the parameters defines properties for surfaces of type -1 (note that properties of type 0 are defined in laura_namelist_data), the second instance defines properties for surfaces of type -2, and so on (see Section 5.3 on page 14 for different solid surface types.)

The parameters that can be defined for each solid surface boundary are:

blowing_model

See Section 5.4.1 on page 17 for more details and a complete list of options.
Default: 'none'

[19] The species_transp_data_0 file should only be changed by developers.

`catalysis_model`

See Section 5.4.12 on page 29 for more details and a complete list of options. Default: `'super-catalytic'`

`char_density`

Density of the char, kg/m^3. Default: 256.29536, which is for the heritage AVCOAT.

`CHONSi_frac_char`

See Section 5.4.1 on page 17 for more details and a complete list of options. Default: `CHONSi_frac_char = 1.0, 0.0, 0.0, 0.0, 0.0`

`CHONSi_frac_pyrolysis`

See Section 5.4.1 on page 17 for more details and a complete list of options. Default: `CHONSi_frac_pyrolysis = 1.0, 0.0, 0.0, 0.0, 0.0`

`emiss_a, emiss_b, emiss_c, emiss_d`

Real numbers to assume emissivity constant coefficients for solid surface boundary type (see Section 5.3 on page 14) surface emissivity, ϵ, which is calculated as

$$\epsilon = \epsilon_a + \epsilon_b T_w + \epsilon_c T_w^2 + \epsilon_d T_w^3 \tag{43}$$

where T_w is the surface temperature. Values for ϵ_a, ϵ_b, ϵ_c, and ϵ_d are defined by `emiss_a`, `emiss_b`, `emiss_c`, and `emiss_d`, respectively. Default: `emiss_a=0.89`, `emiss_b=0`, `emiss_c=0`, and `emiss_d=0`

`h_ablation`

See Section 5.4.1 on page 17 for more details and a complete list of options. Default: `0.0`

`mdot_pressure`

See Section 5.4.1 on page 17 for more details and a complete list of options. Default: `0.0`

`surface_group_name`

Character descriptor for surfaces with solid surface boundary types—see Section 5.3 on page 14. Any character can be specified to group solid surface boundaries. Default: `'undefined surface'`

`surface_temperature`

Initial wall temperature in K for the solid surface boundary. This variable is similar to `twall_bc` in Section 5.4.12 on page 29. The wall temperature stays constant as specified by this parameter if `surface_temperature_type = 'constant'`. Default: value of `twall_bc`.

`surface_temperature_type`

A character identifier for surface temperature model. See Section 5.4.12 on page 29 for options and their descriptions. Default: `'constant'`

`t_ablation`

See Section 5.4.1 on page 17 for more details and a complete list of options. Default: `0.0`

`virgin_density`

Density of virgin material, kg/m^3. Default: 544.627742, which is for the heritage AVCOAT.

6 Output Files

LAURA generates the files listed in Table 2 regardless of inputs specified. A description of each output file is presented after a brief discussion of `stdout` output.

Table 2: LAURA output files.

Filename	Content	Format
`laura.g.fvbnd`	Boundary types in `laura.g`	FieldView™
`laura.nam`	Variables names in `laura.q`	Tecplot™
`laura.q`	Flowfield solution	PLOT3D
`laura_blayer.dat`	Wall and Boundary layer edge data	Tecplot™
`laura_conv.out`	Time step, CPU time, and residuals	ASCII
`laura_new.g`	Volume grid	PLOT3D
`laura_new.rst`	Flowfield solution for restart	Binary
`laura_surface.g`	Surface grid	PLOT3D
`laura_surface.nam`	Variables names in `laura_surface.q`	Tecplot™
`laura_surface.q`	Surface solution	PLOT3D

In addition to files, LAURA writes some information on the screen. After initial configuration information such as grid block sizes, `laura_namelist_data` specifications, and `tdata` gas physics, the screen output is divided to five distinctive categories: block number, task number, L_∞ including location and equation, and L_2 of mixture continuity, xyz-momenta, and energy equations.

The L_2 norm is defined as

$$L_2 = \sum_{i=1}^{N} \left(\frac{|R_i|}{\rho_i} \right)^2 \tag{44}$$

where N is the total number of grid cells times the number of conservation equations, R is the residual, and ρ is the mixture density.

The location of the maximum L_∞ residual is shown by four integer numbers after the norm itself. The first three integer numbers correspond to i-, j-, and k-indices of the corresponding block, and the last number is the equation number, `n_eqn`. The total number of equations, `n_eqn_max`, is defined as

$$\texttt{n\_eqn\_max} = \texttt{n\_species} + 3 + \texttt{n\_energy} \tag{45}$$

where `n_species` and `n_energy` are number of species and energy equations, respectively. The equation number corresponding to species is based on the species order defined in `tdata` followed by the x-, y-, and z-momentum equation numbers. For example, consider block 11 for the following sample screen output,

```
 Reading block sizes from laura.g
  block =   10 task =  10  Linf: 0.63E+02  23  24  86  1  L2eq: 0.31E+00 0.11E-01 0.12E-01 0.12E-01 0.32E-01 0.27E-02
> block =   11 task =  11  Linf: 0.12E+04  21   2  86 14  L2eq: 0.42E+01 0.10E+01 0.17E+00 0.95E+00 0.88E+00 0.28E-01 <
```

```
block =   12 task =  12  Linf: 0.31E+03  11   7  87  5  L2eq: 0.20E+01 0.38E+00 0.22E+00 0.28E+00 0.35E+00 0.11E-01
block =    6 task =   6  Linf: 0.68E+02   6  24  87  1  L2eq: 0.39E+00 0.17E-01 0.80E-02 0.28E-01 0.37E-01 0.32E-02
block =    2 task =   2  Linf: 0.66E+02   3  24  87  1  L2eq: 0.49E+00 0.27E-01 0.88E-02 0.53E-01 0.53E-01 0.38E-02
block =    4 task =   4  Linf: 0.17E+03  24   1  87 14  L2eq: 0.89E+00 0.12E+00 0.47E-01 0.14E+00 0.12E+00 0.75E-02
block =    9 task =   9  Linf: 0.65E+02  21  24  86  1  L2eq: 0.43E+00 0.26E-01 0.36E-01 0.47E-01 0.48E-01 0.32E-02
block =   14 task =  14  Linf: 0.78E+02   2  24  86  1  L2eq: 0.66E+00 0.63E-01 0.27E-01 0.22E-01 0.67E-01 0.36E-02
block =    7 task =   7  Linf: 0.71E+03  14   2  87 14  L2eq: 0.27E+01 0.51E+00 0.11E+00 0.69E+00 0.55E+00 0.22E-01
block =    1 task =   1  Linf: 0.22E+03  24   4  87 14  L2eq: 0.11E+01 0.13E+00 0.29E-01 0.14E+00 0.13E+00 0.90E-02
block =    8 task =   8  Linf: 0.26E+03   6   1  87 14  L2eq: 0.10E+01 0.14E+00 0.97E-01 0.19E+00 0.16E+00 0.82E-02
block =    5 task =   5  Linf: 0.78E+02   1  24  87  1  L2eq: 0.47E+00 0.22E-01 0.23E-01 0.52E-01 0.50E-01 0.35E-02
block =    3 task =   3  Linf: 0.45E+03  24  16  87 14  L2eq: 0.21E+01 0.35E+00 0.80E-01 0.41E+00 0.34E+00 0.18E-01
block =   13 task =  13  Linf: 0.15E+03  17   2  87  5  L2eq: 0.98E+00 0.73E-01 0.63E-01 0.65E-01 0.88E-01 0.39E-02
step  =   10  time =    57.20 sum(abs(task error)) =  0.28E+02 L2 norm =   0.17E-02
```

which is for 14-block, 11-species, two-temperature case. The maximum change is shown to be from z-momentum equation `n_eqn`=14 located on i=21, j=2, and k=86 of block 11.

See Section 7.6 on page 61 for a description of the `laura_stdout_to_tec` utility, which can convert this output into a Tecplot™-compatible format.

6.1 laura.g.fvbnd

FieldView™ boundary data file used to label boundary condition types contained in `laura.g` file. Tecplot™ automatically detects this file when the `laura.g` is loaded.

6.2 laura.nam

Tecplot™ name file used to label variables contained in `laura.q` file.

6.3 laura.q

PLOT3D function file for post-processing volume solution. Most of the variables in this file are non-dimensionalized according to Table 3. The actual number of variables in this file depends on the condition specified in the input files.

Table 3: laura.q variables

Variables	**Definition**	**Normalized by**
$c_{N,N2,...}$	Species mass fraction	-
u, v, w	Velocity components	U_∞
E	Total energy per unit mass	U_∞^2
e_j	Energy mode j per unit mass	U_∞^2
T	Translational temperature	-
T_v	Vibrational temperature	-
ρ	Density	ρ_∞
M_w	Molecular weight	-
p	Pressure	$\rho_\infty U_\infty^2$
c	Sonic velocity	U_∞
e	Static energy per unit mass	U_∞^2
e_v	Vibrational energy per unit mass	U_∞^2

6.4 laura_blayer.dat

Boundary layer edge, and wall surface properties and shock stand off distance are written in this ASCII and Tecplot™ readable file. Below is an example showing the header of the file with their orders. Note that the species mass fraction will be written in the exact same order as provided by user in `tdata` file—see Section 5.5 on page 36. Also, the vibrational temperature will only be written if two temperature is selected in `tdata`.

In addition to these properties, angle-of-attack, Mach and Reynolds-number-per-grid-unit will be provided in this file as auxiliary parameters.

```
TITLE ="BL EDGE PROPERTIES"
VARIABLES = "xw (m)"
"yw (m)"
"zw (m)"
"rhow (kg/m^3)"
"pw (Pa)"
"Tw (K)"
"Tvw (K)"
"Hw (J/kg)"
"muw (Pa.s)"
"c<sub>N2</sub>w"
"c<sub>O2</sub>w"
"c<sub>N</sub>w"
"c<sub>O</sub>w"
"c<sub>NO</sub>w"
"qw (W/m^2)"
"tauwx (Pa)"
"tauwy (Pa)"
"tauwz (Pa)"
"re-cell"
"rhoe(kg/m^3)"
"pe (Pa)"
"Te (K)"
"Tve (K)"
"He (J/kg)"
"ue (m/s)"
"ve (m/s)"
"we (m/s)"
"Me"
"mue (Pa.s)"
"c<sub>N2</sub>e"
"c<sub>O2</sub>e"
"c<sub>N</sub>e"
"c<sub>O</sub>e"
"c<sub>NO</sub>e"
```

```
"delta (m)"
"deltastar (m)"
"theta (m)"
"Re-ue (1/m)"
"CH (kg/m^2.s)"
"Stand-off (m)"
DATASETAUXDATA Common.AngleOfAttack="-70.000"
DATASETAUXDATA Common.ReferenceMachNumber=" 7.130"
DATASETAUXDATA Common.ReynoldsNumber="  4037.    "
ZONE T="laura_blayer: Block  1"
..
..
```

6.5 laura_conv.out

Time steps and residuals history are written in this file. Some of the flow conditions, such as angle-of-attack, free stream conditions, Mach number, Reynolds-number-per-grid-unit, etc. are also repeated at the beginning of the file. As shown in the following sample, step number, clock time, sum of all the residuals in all active tasks, and the overall L_2 norm of the residuals are written in this file. The overall L_2 norm is defined as:[20]

$$L_2 = \frac{\sum_i (|rhs_i|/\rho_i)^2}{CFL^2} \tag{46}$$

where rhs is the residual and ρ is the density.

```
Step      0 time=     4.53
step  =   10  time =    10.87 sum(abs(task error)) =  0.27E-03 L2 norm =   0.92E-12
step  =   20  time =    17.13 sum(abs(task error)) =  0.26E-03 L2 norm =   0.88E-12
step  =   30  time =    23.40 sum(abs(task error)) =  0.25E-03 L2 norm =   0.84E-12
step  =   40  time =    29.65 sum(abs(task error)) =  0.25E-03 L2 norm =   0.81E-12
  .
  .
```

See Section 7.5 on page 61 for a description of the `laura_conv_to_tec` utility, which can convert this file into a Tecplot™-compatible format.

6.6 laura_new.g

The PLOT3D grid file includes any changes associated with grid adaptation or grid doubling during the run. The name must be changed to `laura.g` if user wants to restart with this new grid on the next run—see Section 5.2 on page 14 for more information on the file format.

6.7 laura_new.rst

The restart file contains volume and surface data required for restart from end of current run. The name must be changed to `laura.rst` if user wants to restart from

[20] The overall L_2 norm definition is different than the L_2 norm defined for each equation by Equation 44.

this new solution file.

6.8 laura_surface.g

PLOT3D 3d whole multiblock surface grid file. See Section 5.2 on page 14 for more information on the file format.

6.9 laura_surface.nam

Tecplot™ name file used to label variables contained in `laura_surface.q` file.

6.10 laura_surface.q

PLOT3D function file for post-processing surface solution. The number of variables in this file depends on the conditions specified in the `laura_namelist_data`. Some of the surface variables are presented in Table 4.

Table 4: laura_surface.q variables.

Variables	**Definition**	**Unit**
T	Surface temperature	K
p	Surface pressure	N/m^2
τ_x, τ_y, τ_z	Wall shear stresses	N/m^2
q_{conv}	Convective heat flux	W/cm^2
q_{rad}	Radiative heat flux	W/cm^2
ϵ	Surface emissivity	-
$mdot$	Blowing or Suction rate	kg/m^2-s

7 Laura Utilities

LAURA has several interactive utilities that automatically generate some of the required input files or otherwise aid running and post-processing LAURA simulations. These utilities are explained here:

7.1 bounds

This interactive utility creates `laura_bound_data` that contains block connectivity data. This utility reads the volume grid data from a PLOT3D structured grid, `laura.g`. See Section 5.2 on page 14 for more detail on the file format. Here is a sample of an interactive session with `bounds`:

```
 Enter precision of laura.g : 1 = single, 2 = double
2
 Do you want all type 9 bounds to default to type 8?
 Enter 1 for yes or 0 for no: (0)
0
 .
 .
 BLOCK  =    1 k  =  1 BOUNDARY

 Area  =   0.26821087E+03
 (Xcntr,Ycntr,Zcntr) = (-0.63020306E+02, 0.91387367E+02,-0.63128987E+03)

 Enter ITYPE: (0)
0
 .
 .
```

The first question is about the precision of `laura.g`. Note that any grid file created by LAURA or one of its utilities is created in double-precision.

The second question is about type 8 and type 9 boundary data. These boundary types are given for the block faces that are shared between two blocks. A type 9 requires an identical orientation of indices across the shared boundary (increasing i to increasing i, increasing j to increasing j, and increasing k to increasing k). A type 8 accommodates more general connectivity.

7.2 coarsen

Use this utility to coarsen the grid, `laura.g`, and solution, `laura.rst`, files in i-, j-, and/or k-directions. The new files, `laura_new.g`, and `laura_new.rst`, will be double precision regardless of the input grid precision. This utility does not accept single precision `laura.rst`.

7.3 convert_bound_data

The utility converts old (pre LAURA.5) `STRTfiles/bound_data.strt` files to the new `laura_bound_data` format. Usage: `convert_bound_data bound_data.strt`.

7.4 convert_laura

This interactive utility converts cases run with prior versions of LAURA.5—see Appendix A on page 75. This utility generates `laura.g`, from `old.rst`, and a new restart, `laura_new.rst`.

The following file are either required or optional prior to the execution of this utility:

`old.2eq`

This file, which may have a different root name, has two-equation turbulent model information. This file is optional.

`old.ep+`

This file, which may have a different root name, has algebraic turbulent model information. This file is optional.

`old.qtw`

This file, which may have a different root name, has surface temperature information. This file is optional. If this file is provided, free stream density and velocity are also needed. LAURA uses this information to approximately calculate the related parameters needed to be in `laura_new.rst`.

`old.rst`

This file, which may have a different root name, is the old restart file that contains volume and surface data. The utility will ask for the precision of the data and the numbers of grid-blocks that are in this file. This file is required.

`laura_bound_data`

Use the `convert_bound_data` utility, or see Appendix A on page 75 to convert this file from old format. This file is required.

7.5 laura_conv_to_tec

This utility translates the LAURA convergence history file, `laura_conv.out`, described in Section 6.5 on page 58 into a form usable by Tecplot™.
Usage: `laura_conv_to_tec [options]`.

7.6 laura_stdout_to_tec

To allow task-specific convergence history plotting, this utility translates the LAURA standard output stream described in Section 6 on page 55 into a convergence history file for each task suitable for Tecplot™.
Usage: `laura_stdout_to_tec [options] [file]`.

7.7 make_assign_tasks

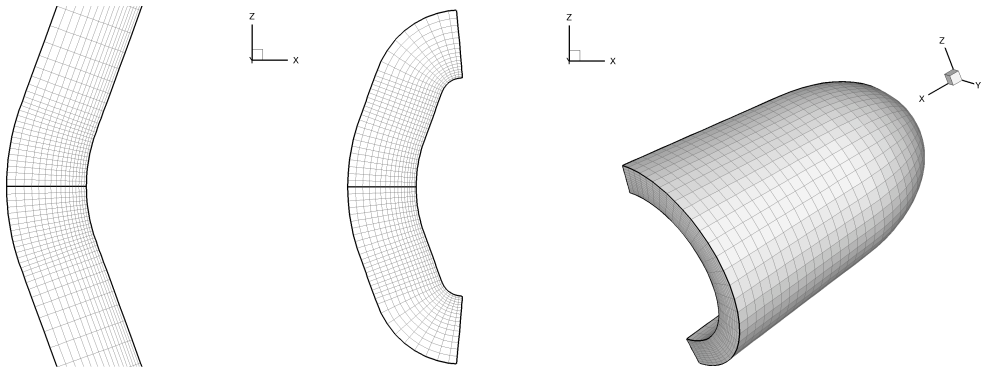
Given the number and blocks and processes, the `make_assign_tasks` utility will generate a default `assign_tasks` (point relaxation with k-sweeps).
Usage: `make_assign_tasks n_blocks n_processes`.

7.8 prolongate

Use this utility to generate fine adapted grid, `laura_new.g`, and solution, `laura_new.rst`, files from coarse adapted grid, `coarse.g`, and solution, `coarse.rst`, files. You will need to provide a fine un-adapted grid filename and coarsening stride used in the `coarse.g` file.

7.9 self_start

This interactive utility generates a single-block structured grid, `laura.g`, for families of 2D, axisymmetric and 3D blunt bodies. This utility will also generate `laura_bound_data`. Schematics of several blunt body families are shown in Figure 3. Note that for axisymmetric geometries, the symmetry boundaries must be on the y-axis. Parameters for defining a probe shape are shown in Figure 4 on the facing page.



(a) Spherically blunted cone. (b) Custom aerobrake. (c) Asymmetric 3D cone.

Figure 3: Sample geometries generated by `self_start` utility.

7.10 shuffle_laura

This interactive utility modifies (shuffles) variables in the LAURA restart, `laura.rst`, to enable continuation of simulation if the gas model variables or parameters including number of species, thermal nonequilibrium, radiation, ablation, and turbulence model are modified. The users will be prompted to provide necessary information.

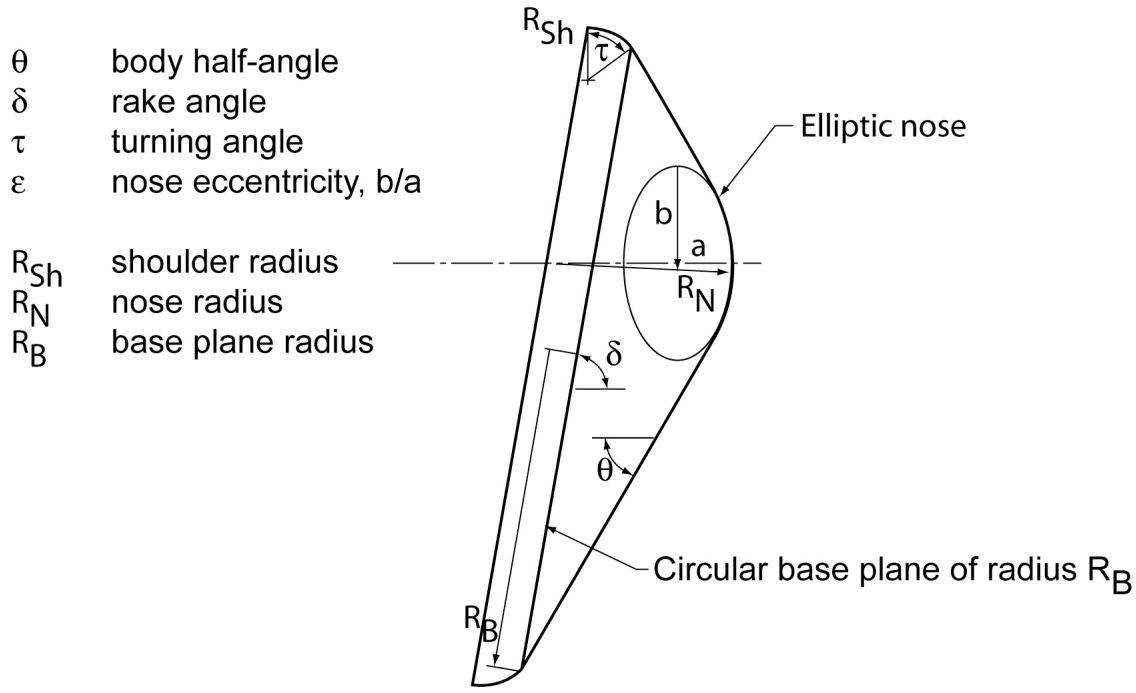


Figure 4: Definition of probe shape parameters used by `self_start`.

8 Sample Cases

8.1 Sphere: 5-species Air, Thermo-chemical Nonequilibrium

A working directory is created in which all requisite files are assembled. The utility `self_start` is used to generate a grid and the initial template for some required input files within this working directory. A verbatim transcript of an interactive session using `self_start` follows.

```
% self_start
 Select dimensionality:
 1 = Axisymmetric
 2 = Two-dimensional
 3 = Three-dimensional
1
 Select geometry:
 1 = Conic (cone/wedge, paraboloid, etc.).
 2 = Aerobrake (includes AFE without axis singularity).
2
 Select aerobrake type:
 0 = AFE
 1 = hemisphere
 2 = customized aerobrake
1
Enter radius (m) {     1.000000}:
1.
 Select number of cells along symmetry plane.
30
 Enter number of cells in k direction, prior to any doubling
16
```

At this point the grid file `laura.g`, the boundary data file `laura_bound_data`, the task assignment file `assign_tasks`, and the namelist file `laura_namelist_data` are created. The boundary data file does not require any further modification. The task assignment file is set for point-implicit relaxation - the standard practice for starting any simulation. The namelist file requires editing to define free stream conditions and possibly alter default settings. The edited file used for the first run of the test case follows.

```
&laura_namelist
velocity_ref = 5000. ! reference velocity, m/s
density_ref  = 0.001 ! reference density, kg/m^3
tref         = 200.0 ! reference temperature, K
alpha        = 0.000 ! pitch angle, degrees
twall_bc     = 500.0 ! initial wall temperature, K
chem_flag  = 1  ! 0 chemically frozen, 1 chemical source on
```

```
therm_flag = 1  ! 0 thermally frozen,  1 thermal source on
irest    = 0    ! 0 for fresh start, 1 for restart
ncyc     = 2000 ! global steps
jupdate  = 4    ! steps between update of jacobian
ntran    = 4    ! steps between update of transport properties
nitfo    = 1500 ! number of 1st-order relaxation steps
iterwrt  = 200  ! steps between saves of intermediate solution
rf_inv   = 3.0  ! inviscid relaxation parameter
rf_vis   = 1.0  ! viscous relaxation parameter
movegrd  = 100  ! number of steps between calls to align_shock
maxmoves = 0    ! maximum number of calls to align_shock
re_cell  = 0.1  ! target cell Reynolds number at wall
fsh      = 0.6  ! target bow shock position arc length fraction
kmax_error = 0.005 ! error norm for doubling grid in k-dir.
kmax_final = 64 ! max number cells in k dir after all doubling
nexch    = 2    ! steps between exchange of info in mpi
frac_line_implicit = 0.7 ! fraction of line by block tri-dia
surface_temperature_type_0 = `radiative equilibrium'
catalysis_model_0 = `super-catalytic'
emiss_a_0 = 0.89
ept = 0.010     ! relaxation factor on read eq wall bc
axi_symmetric = .true.
two_dimensional = .false.
xmc =   0.0000
ymc =   0.0000
zmc =   0.0000
grid_conversion_factor =   1.0000
sref =  0.43633E-01
cref =   2.0000
/
```

The first 5 variables of the namelist in this particular template (other variable orderings are acceptable) deal with free stream conditions. The user must set these values, otherwise the user will get the default values assuming laminar flow of a perfect gas at 5 km/s and 0.001 kg/m^3. In this case both the `chem_flag` and `therm_flag` are reset to 1 to turn on the chemical and thermal source terms. Other template values are defined to provide a reasonable compromise between solution robustness and convergence rate for a fresh start solution (irest = 0). The template calls for 2,000 relaxation steps (`ncyc`) in the initial run with jacobian updates (`jupdate`) and transport property updates (`ntran`) requested every four relaxation steps. The first 1,500 iterations are executed using first-order spatial accuracy (`nitfo`)—second-order accuracy does not contribute significantly to the solution evolution in the initial relaxation period. The inviscid and viscous relaxation factors (`rf_inv` = 3 and `rf_vis` = 1) multiply the respective contributions to the Jacobian matrices and provide damping of the update. Larger values sometimes improve robustness for more energetic flows but are not required in this case.

Template values for grid movement are `movegrd` = 100, `maxmoves` = 0 (unlimited number of moves), `re_cell` = 0.1, and `fsh` = 0.6. These values are approximately tuned for optimal response in the opening relaxation process from a fresh start where the body materializes in a supersonic flow. Allowing the grid to move every 100 steps provides frequent opportunity to follow the evolution of the shock front as it initially is collapsed on the surface and then reflects off the surface into the oncoming flow. Setting `re_cell` to 0.1 provides very tight stretching near the wall. If there is a large difference between the wall temperature and the temperature of the first cell center off the wall then the upwind algorithm may fail to sense the wall - the boundary condition using the Roe's averaged interface may admit a supersonic flow directed toward the wall at the interface. This condition subverts the ability of the inviscid, no-slip boundary to properly engage. The tighter near wall resolution enables the upwind scheme to sense the wall under all wall temperature conditions tested to date. Setting `fsh` = 0.6 targets the captured bow shock location at 60% of the distance between the wall and the inflow boundary, providing adequate margin for the shock to reflect outward without striking the inflow boundary prior to the next grid update.

Automated grid doubling in the k direction is controlled by the triggering error norm magnitude (`kmax_error` = 0.005) and the maximum number of cells in the k direction (`kmax_final` = 64). Recall that the grid was initialized with only 16 cells in the k direction with the `self_start` utility. The grid will double to 32 cells in the k direction (normal to the wall) when the L2 norm first drops below 0.005. It will double again when the error norm next falls below this trigger point. The selection of a trigger point for more complex problems (three-dimensional, highly energetic) may require user experimentation: trigger too high and the grid doubles too early causing a lot of extra work; trigger too low and the solution may ring (limit cycle) on a coarse grid and never engage a finer grid.

The template setting therefore updates the boundary condition after completion of a forward and backward sweep through the domain. Other template values are not discussed here. They are consistent with default values as described in Section 5.4 and are included for the users convenience.

Gas property data files must now be copied (or linked) from the `share/physics_modules` installation directory into the working directory prior to executing laura. The files include `species_thermo_data` (thermodynamic curve fit data), `species_transp_data` and `species_transp_data_0` (collision cross section data for computing transport properties), `kinetic_data` (chemical kinetic source term data) and `tdata` (the gas model specific to the current simulation). The `tdata` file is generally the only file that requires editing by the user. For the case of 5 species air in thermochemical nonequilibrium the file `tdata` is defined as follows.

```
Two Temperature
N  6.217e-20
O  7.758e-09
N2  0.737795
O2  0.262205
NO  1.e-09
```

The fresh start run for this example case is now ready to be executed. It is assumed that the executable file laura is available in the working directory. This case is small enough to be run in interactive mode. For the purposes of discussion, it is convenient to capture the output in a file called `out_01`.

```
% laura >& out_01       (for csh)
% laura > out_01 2>&1   (for sh)
```

The user should note several types of information in `out_01`. The beginning of this file contains diagnostic information regarding presence of optional files, free stream conditions, and kinetic model diagnostics including warnings regarding absence of some allowed third body collision partners. Lines beginning with "step =" keep track of the relaxation step number, elapsed wall time, sum of the L1 norms over all conservation equations, and the L2 norm. Lines beginning with "block =" keep track of the L1 norm for mixture continuity, x-momentum, y-momentum, z-momentum, total energy, and vibrational-electronic energy residuals. The statement `Calling align_shock...` appears after every 100 steps (`movegrd` = 100). The statement `Saving restart and plot files.` followed by intermediate values of aerodynamic coefficients appear after every 200 steps (`iterwrt` = 200). The grid doubles automatically after step 472 from 16 to 32 cells in the k direction when the error norm first drops below 0.005.

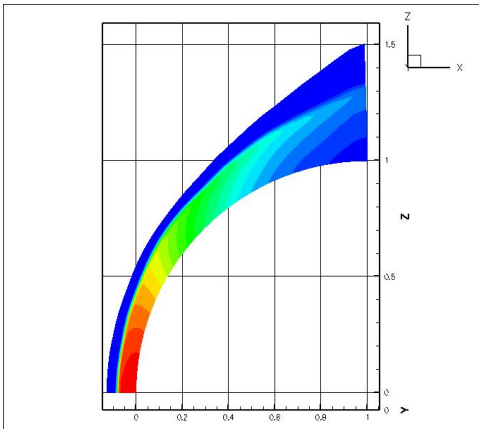
```
block =    1  task =   1 err:  0.17E+01  0.30E+00  0.61E-15  0.44E+00  0.38E+00  0.53E-01
step  =  472  time =    24.65 sum(abs(task error)) =  0.29E+01 L2 norm =   0.50E-02

Increased kmax to  32
block =    1  task =   1 err:  0.69E+05  0.54E+02  0.31E-14  0.51E+02  0.14E+04  0.40E+03
step  =  476  time =    25.09 sum(abs(task error)) =  0.70E+05 L2 norm =   0.97E+08
```

In general, there is a large jump in the error norm following a grid move or grid doubling which rapidly diminishes to pre-adjustment levels. A second doubling from 32 to 64 cells occurs after step 1348.

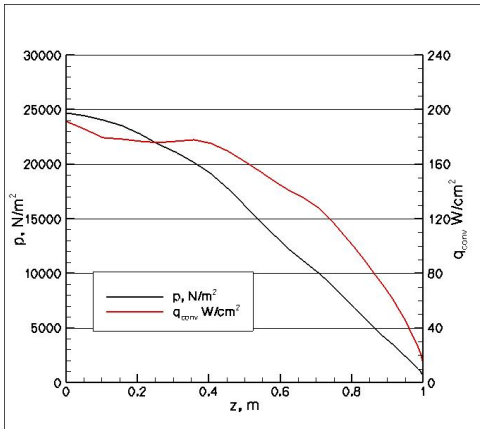
```
block =    1  task =   1 err:  0.15E+01  0.47E+00  0.13E-14  0.34E+00  0.45E+00  0.47E-01
step  = 1348  time =   113.30 sum(abs(task error)) =  0.28E+01 L2 norm =   0.49E-02

Increased kmax to  64
block =    1  task =   1 err:  0.47E+05  0.46E+02  0.39E-14  0.41E+02  0.82E+03  0.23E+03
step  = 1352  time =   114.19 sum(abs(task error)) =  0.49E+05 L2 norm =   0.46E+08
```

The interim solution for pressure contours after the first 2,000 steps is shown in Figure 5a on the next page. The corresponding surface pressure and heating distributions are shown in Figure 5b. The deep blue zone in front of the sphere represents undisturbed free stream conditions. The deep red indicated the high pressure stagnation region. The captured shock is approximately located at 60% of the distance between the spherical surface and the inflow boundary. The surface pressures and shock shape will be shown to be nearly converged at this point but the heating rates are still far from converged. Converging the boundary layer profile is the focus of the remaining relaxation steps.

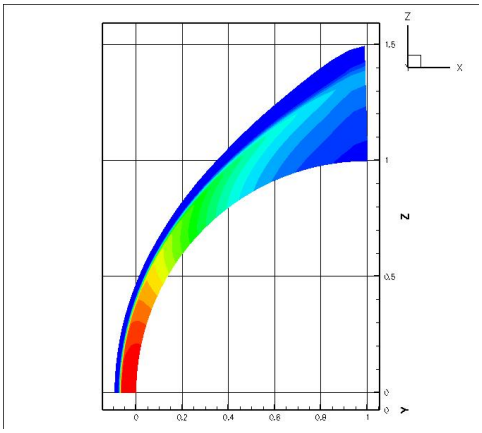
Line relaxation is engaged at this point by editing `assign_tasks`, sweeping around the sphere in the i direction and applying line relaxation across the boundary layer in the k direction.



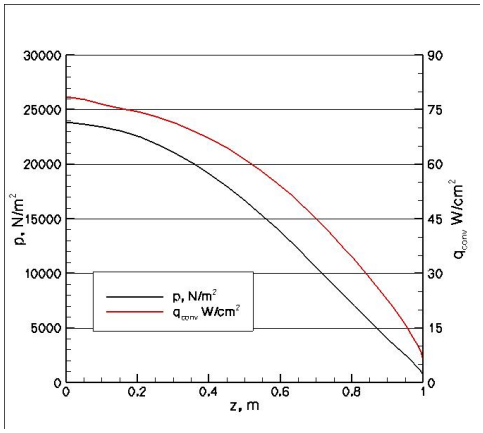
(a) Pressure contours - interim solution after 2000 steps



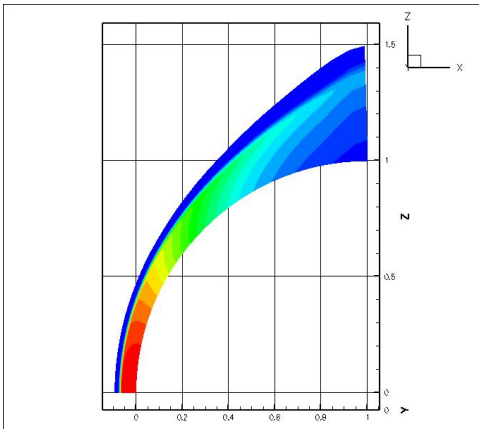
(b) Surface pressure and heating - interim solution after 2000 steps



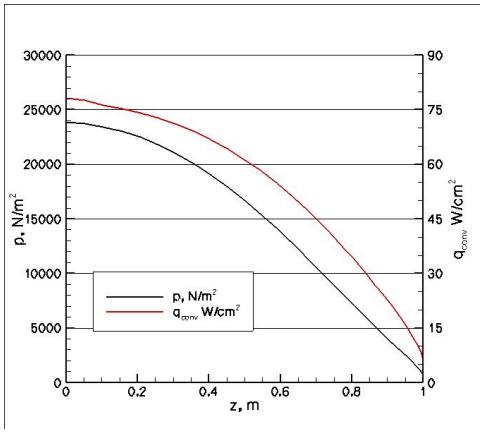
(c) Pressure contours - converged solution after 4000 steps.



(d) Surface pressure and heating - converged solution after 4000 steps.



(e) Pressure contours - 1.e-12 error norm solution after 5600 steps.



(f) Surface pressure and heating - 1.e-12 error norm solution after 5600 steps.

Figure 5: Solutions to the sphere test problem: $V_\infty = 5{,}000$ m/s, $\rho_\infty = 0.001$ kg/m^3, $T_\infty = 200$ K, 5-species air, thermochemical nonequilibrium, radiative equilibrium wall, $\epsilon = 0.89$, and super-catalytic wall boundary.

```
1    1    3    1    0    1
```

The adapted grid and restart files are renamed to start the second run.

```
% cp laura_new.g laura.g
% cp laura_new.rst laura.rst
```

Changes or additions to the file `laura_namelist_data` are indicated below.

```
irest   = 1 ! 0 for fresh start, 1 for restart
jupdate = 20 ! steps between update of jacobian
ntran   = 20 ! steps between update of transport properties
nitfo = 0 ! number of 1st-order relaxation steps
rf_inv = 2.0 ! inviscid relaxation parameter
rf_vis = 2.0 ! viscous relaxation parameter
movegrd = 400 ! number of steps between calls to align_shock
maxmoves =  3 ! maximum number of calls to align_shock
frac_line_implicit = 0.7
```

At the conclusion of these next 2,000 steps (4,000) total the L2 norm has dropped to 0.74e-06 and the solution is converged. The solution for pressure contours after 4,000 steps is shown in Figure 5c. The corresponding surface pressure and heating distributions are shown in Figure 5d. Significant reduction in heating level and smoothing of the stagnation region profile has occurred using the line relaxation across the boundary layer during this second set of 2,000 relaxation steps.

Line relaxation across the entire shock layer (`frac_line_implicit` = 1.0) can be accommodated in this case if the relaxation factors (`rf_inv` and `rf_vis`) are increased to 5. The tolerance for convergence of the L2 norm (rmstol) is set to 1.e-12. Grid movement is switched off (movegrd = 0). The latest grid and restart files are renamed as before to start the third run. The third run reaches the convergence criteria in 1,600 additional relaxation steps, a drop of 6 orders of magnitude in the L2 norm. The solutions (Figure 5e and Figure 5f) are nearly identical to the corresponding figures at 4,000 steps.

```
block =    1  task =   1 err:  0.14E-04  0.50E-05  0.11E-14  0.36E-05  0.40E-05  0.53E-06
step  = 1580  time =   335.39 sum(abs(task error)) =  0.27E-04 L2 norm =   0.10E-11

block =    1  task =   1 err:  0.14E-04  0.48E-05  0.12E-14  0.35E-05  0.39E-05  0.51E-06
step  = 1600  time =   339.63 sum(abs(task error)) =  0.26E-04 L2 norm =   0.98E-12

Aerodynamic Coefficients
c_x = -0.8854
c_y = -0.0000
c_z =  0.6950
c_l = -0.0000
c_m =  0.3514
c_n =  0.0000
```

8.2 Coupled radiation procedure

Starting with the converged non-radiating LAURA solution, the `shuffle_laura` routine must be applied to the `laura.rst` file, and the option to convert from uncoupled to coupled radiation must be chosen—see Section 5.4.11 on page 28 for

more info on radiation flags. The new .rst file created by shuffle_laura must then be renamed to laura.rst. Furthermore, the following lines must be added to the laura_namelist_data file:

```
radiation = .true.
nrad = 5000
frac_rad_new = 0.8
iinc_rad = 3
jinc_rad = 3
```

The first of these, radiation, must be set to true. The parameter nrad specifies the number of flowfield iterations between calls to HARA, frac_rad_new specifies the fraction of the most recent HARA calculation applied to the LAURA solution, and iinc_rad and jinc_rad, respectively, specify the increment in the treated points along the surface and in the spanwise direction. For axisymmetric cases, jinc_rad must be 1. For relatively weakly coupled radiation, such as Earth lunar-return conditions [22], nrad may be set to 1,500 for the first two calls to HARA, and 3,000–5,000 for the subsequent calls. For strongly coupled flows, such as Mars-return entry to Earth [11], nrad should be set to 500 for the first two calls to HARA, and 1,000 for the rest.

The radiation models applied by the radiation code (HARA) are defined by the optional file hara_namelist_data described in Section 5.6 on page 38. If this file is not present in the working directory, then the code determines which radiative mechanisms to apply based on the species number densities in the flowfield.

8.3 Unspecified ablation procedure - Coupled

The recommended procedure for an unspecified ablation computation, meaning the ablation rate and wall temperature is computed as part of the flowfield solution (instead of being specified by the user), is as follows:

1: Obtain a non-ablating solution assuming an equilibrium catalytic and radiative equilibrium wall. Include only species required for a non-ablating solution.

2: Apply the shuffle_laura utility to the converged non-ablating solution. Choose the ablation option and increase the number of species to the amount required to accommodate ablation species. Add the ablation species to tdata.

3: Modify laura_namelist_data to include the following—see Section 5.4.1 on page 17 for more info on ablation parameters:

```
surface_temperature_type_0 = 'surface energy balance'
blowing_model_0 = "equil_char_quasi_steady"
CHONSi_frac_pyrolysis_0 = 0.547, 0.093, 0.341, 0.019, 0.000
CHONSi_frac_char_0 =      0.488, 0.000, 0.273, 0.000, 0.239
ept = 0.01
nexch    = 2
```

```
freq_wall = 50
bprime_flag = 1
compute_mdot_initial = 1
ablation_option = 0
ablation_verbose = .true.
```

where `CHONSi_frac_pyrolysis_0` and `CHONSi_char_pyrolysis_0` should be changed to represent the material of interest. Setting `bprime_flag = 1` specifies that an approximate film coefficient diffusion model is applied in the surface elemental mass balance. This model is robust enough to apply to a converged non-ablating flowfield. For this option, `freq_wall` specifies how often the cells at the wall are updated (instead of `nexch`). A value of 50 seems to work for both weakly and strongly ablating cases. In addition, `ept` represents the fraction of the new ablation solution, which includes the ablation rate, wall temperature, and wall species.

4: Run LAURA for roughly 24,000 iterations. During each ablation computation, data is printed out for each point on the body, indicated by `l`. The level of convergence is indicated with `mdot residual` at the end of ablation computation:

$$\texttt{mdot residual} = \sum_{l} (\Delta \dot{m})^2 \tag{47}$$

Usually, `mdot residual = 1.E-2` or lower indicates that ablation computation is adequately converged within 1%.

5: If the user considers the b-prime approach of sufficient accuracy, then the computation is finished. If the user wishes to apply a rigorous diffusion model at the surface, consistent with that applied throughout the flowfield, then the following modifications should be made to `laura_namelist_data`:

```
freq_wall = 500
bprime_flag = 0
```

Setting `bprime_flag = 0` specifies the rigorous diffusion model at the surface [2]. This model is significantly less robust than the b-prime approach (`bprime_flag = 1`) , which is why it requires the solution of the b-prime approach as an initial condition. With `bprime_flag = 0`, the energy equation is solved separately from the elemental mass balance and char equilibrium constraints. The number of flowfield iterations between solutions of the energy equation is governed by `freq_wall`, while the other equations are governed by `nexch`. In general, `freq_wall` should be much greater than `nexch`. The energy equation requires the convective heating, which must be allowed to converge to a meaningful value after the wall properties are perturbed. Note that with `bprime_flag = 0`, the ablation calls are significantly quicker than for `bprime_flag = 1`, and nothing is printed to the screen.

6: Run LAURA until the ablation rate, wall temperature, and convective heating reach steady values within one percent. After each solution of the energy

equation, an increase in the residual will be seen. Unlike the b-prime approach, this value can be reduced within a reasonable number of iterations to `1e-7`, or lower.

8.4 Unspecified ablation procedure - Uncoupled

The uncoupled ablation analysis (defined in detail in Ref. [2]) differs from the coupled analysis in that the influence of ablation on convective heating is treated approximately using the blowing correction. In this uncoupled analysis, ablation is never introduced into the flowfield, and it consists of simply a post-processing step to the non-ablating flowfield. The recommended procedure for an uncoupled ablation analysis involves the same first two steps listed in Section 8.3 on page 70 for a coupled analysis. After those steps, modify `laura_namelist_data` to include the following:

```
surface_temperature_type_0 = 'surface energy balance'
blowing_model_0 = "equil_char_quasi_steady"
CHONSi_frac_pyrolysis_0 =   0.8822, 0.0283., 0.0866, 0.0029, 0.0
ept = 1.0
bprime_flag = 1
uncoupled_ablation_flag = 1
ncyc  = 0 ! global steps
```

where `CHONSi_frac_pyrolysis_0` and `CHONSi_char_pyrolysis_0`, should be changed to represent the ablator material of interest. Note that `ncyc = 0` is required. The final step is to run LAURA. This will first call the ablation model. Then it will print out an updated `laura_surface.q` file with the computed ablation rate, wall temperature, and altered convective heating (from the blowing correction) values. LAURA will then terminate without executing any flowfield iterations.

References

1. Mazaheri, A.; Gnoffo, P. A.; Johnston, C. O.; and Kleb, B.: Laura Users Manual: 5.2-43231. NASA TM 215944, Nov. 2009.

2. Johnston, C. O.; Gnoffo, P. A.; and Mazaheri, A.: A Study of Ablation-Flowfield Coupling Relevant to the Orion Heatshield. AIAA Paper 2009–4318, 2009.

3. Stewart, D. A.: Surface Catalysis and Characterization of Proposed Candidate TPS for Access-to-Space Vehicles. NASA TM 112206, July 1997.

4. Scott, C. D.: Catalytic Recombination of Nitrogen and Oxygen on High Temperature Reusable Surface Insulation. *AIAA Progress in Astronautics and Aeronautics: Aerotermodynamics and Planetary Entry*, A. L. Crosbie, ed., AIAA, 1981, pp. 192–213.

5. Zoby, E. V.; Gupta, R. N.; and Simmonds, A. L.: Temperatuure-Dependent Reaction Rate Expression for Oxygen Recombination. *AIAA Progress in Astronautics and Aeronautics: Thermal Design of Aeroassisted Orbital Transfer Vehicles*, H. F. Nelson, ed., AIAA, 1985, pp. 445–464.

6. Cheatwood, F. M.; and Thompson, R. A.: The Addition of Algebraic Turbulence Modeling to Program LAURA. NASA TM 107758, Apr. 1993.

7. Srinivasan, S.; Tannehill, J. C.; and Weilmuenster, K. J.: Simplified Curve Fits for the Thermodynamic Properties of Equilibrium Air. NASA RP 1181, June 1987.

8. Prabhu, R. K.; and Erickson, W. D.: A Rapid Method for the Computation of Equilibrium Chemical Composition of Air to 15,000 K. NASA TP 2792, Mar. 1988.

9. Johnston, C. O.; Hollis, B. R.; and Sutton, K.: Spectrum Modeling for Air Shock-Layer Radiation at Lunar-Return Conditions. *Journal of Spacecraft and Rockets*, vol. 45, no. 6, Nov–Dec 2008, pp. 865–878.

10. Johnston, C. O.; Hollis, B. R.; and Sutton, K.: Non-Boltzmann Modeling for Air Shock-Layer Radiation at Lunar-Return Conditions. *Journal of Spacecraft and Rockets*, vol. 45, no. 6, Nov–Dec 2008, pp. 879–890.

11. Johnston, C. O.; Gnoffo, P. A.; and Sutton, K.: The Influence of Ablation on Radiative Heating for Earth Entry. *Journal of Spacecraft and Rockets*, vol. 46, no. 3, May–June 2009, pp. 481–491.

12. Gally, T.: Development of Engineering Methods for Nonequilibrium Radiative Phenomena about Aeroassisted Entry Vehicles. Ph.D. Thesis, Texas A&M, 1992.

13. Johnston, C. O.; Hollis, B. R.; and Sutton, K.: Radiative Heating Methodology for the Huygens Probe. *Journal of Spacecraft and Rockets*, vol. 44, no. 5, Sep–Oct 2007, pp. 993–1002.

14. *Multi-Species Subsonic Inlet Boundary Condition Formulation with RCS and SRP Applications.*, International Planetary Probe Workshop (IPPW) 7, Barcelona, Spain, 2010.

15. Gordon, S.; and McBride, B. J.: Computer Program for calculation of Complex Equilibrium Compositions and Applications. NASA RP 1311, 1994.

16. Ali, A. W.: The Harmonic and Anharmanic Models fro Vibrational Relaxation and Dissociation of the Nitrogen Molecule. U.S. Navy NRL Memo 5924, Dec. 1986.

17. Millikan, R. C.; and White, D. R.: Systematics of Vibrational Relaxation. *Chem. Phys.*, vol. 39, no. 12, Dec. 1963, pp. 3209–3213.

18. Fisher, B. D.; Holmes, B. J.; and Stough, H. P.: A flight evaluation of a trailing anemometer for low-speed calibrations of airspeed systems on research aircraft. NASA TP 1135, Feb. 1978.

19. Gupta, R.; Yos, J.; Thompson, R. A.; and Lee, K.: A Review of Reaction Rates and Thermodynamic and Transport Properties for an 11-Species Air Model for Chemical and Thermal Nonequilibrium Calculations to 30,000 K. NASA RP 1232, Aug. 1990.

20. Wright, M.: Recommended Collision Integrals for Transport Property Computations Part 1: Air Species. *AIAA J.*, vol. 43, no. 12, 2005, pp. 2558–2564.

21. Wright, M.: Recommended Collision Integrals for Transport Property Computations Part 2: Mars and Venus Entries. *AIAA J.*, vol. 45, no. 1, 2005, pp. 281–288.

22. Gnoffo, P. A.; Johnston, C. O.; and Thompson, R. A.: Implementation of Radiation, Ablation, and Free-Energy Minimization Modules for Coupled Simulations of Hypersonic Flow. AIAA Paper 2009–1399, 2009.

23. Cunto, W.: TOPbase at the CDS. *Astronomy and Astrophysics*, vol. 275, 1993, pp. L5–L8.

Appendix A

Migrating Cases from Prior Versions

Follow the following steps to start LAURA simulations run with LAURA prior to version 5.

Step 1. Create a new working directory and change to it

```
% mkdir [Working Directory]
```

Step 2. Copy following required files from the old directory:

```
% cp /old/root.rst old.rst
% cp /old/root.tsk assign_tasks
% cp /old/STRTfiles/bound_data.strt laura_bound_data
```

and, depending on the case, these files:

```
% cp /old/root.qtw old.qtw
% cp /old/root.2eq old.2eq
```

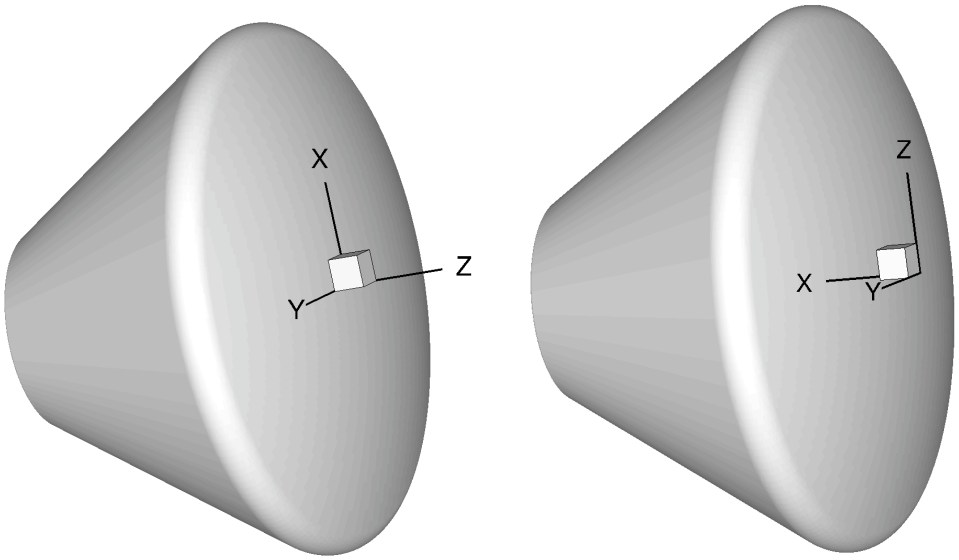
Step 3. Use the LAURA `convert_bound_data` utility, or edit `laura_bound_data` so that there are only integer numbers in this file and there is 6 integers separated by at least one space per line per computational block. For example, the original file may look something like:

```
C:CDATADATADATADATADATADATADATADATADATADATADATADATADATADATADATADATADATA

C:::::: $Name:  $

      data ( itype(i,  1), i=1,6 )
     &          /       1, 1002111,       2,       1,       0,       3 /
      data ( itype(i,  2), i=1,6 )
     &          / 1001211,       1,       2,       1,       0,       3 /

C::CDATADATADATADATADATADATADATADATADATADATADATADATADATADATADATADATADATA
```

Edit the file to look like this:

```
       1  1002111        2        1        0        3
 1001211        1        2        1        0        3
```

Step 4. Run the interactive `convert_laura` utility and answer all the questions. The restart file from LAURA prior to version 5 is single-precision but files produced by LAURA 5 and its utilities are all double-precision. This utility creates `laura.rst` and `laura.g` files. You will have an option either to keep the prior-to-version-5 coordinate system orientation or to rotate the grid



(a) Prior to Version 5. (b) Version 5.

Figure A6: LAURA coordinate system orientations.

to the version 5 coordinate system orientation. As of version 5, LAURA uses $+x$-axis as the body normal direction while prior to version 5 LAURA used the $+z$-axis as the body-normal direction. The two orientations are show in Figure A6.

To run LAURA with LAURA grid orientation prior to version 5, use the following formulas to correct the angle-of-attack α, and the center of the moments coordinates:

$$\alpha_{new} = \alpha_{old} - 90 \tag{A48}$$

$$\begin{aligned} (xmc)_{new} &= -(zmc)_{old} && \text{(A49)} \\ (ymc)_{new} &= +(ymc)_{old} && \text{(A50)} \\ (zmc)_{new} &= +(xmc)_{old} && \text{(A51)} \end{aligned}$$

Step 5. Copy the example `laura_namelist_data` file to your working directory from the `[installs_prefix]/share/laura` directory, where `install_prefix` is the installation prefix specified when LAURA was installed.

```
% cp [install_prefix]/share/laura/laura_namelist_data .
```

Use this file as a template to define the simulation parameters. Refer to Section 5.4 on page 17 for complete list of options. You must change `irest = 0` to `irest = 1`, otherwise the solution will be initialized to free stream conditions.

Step 6. Edit `assign_tasks` so that there are only integer numbers in this file. For example, the original file may look something like this:

```
CASE:
C:::::: $Name:  $

 1  3   0   1    0     1
 2  3   0   1    0     2
blk swp rlx strt stop task
```

Edit the file to read like this:

```
1  3   0   1    0     1
2  3   0   1    0     2
```

Note: you can also use the LAURA utility `make_assign_tasks` to generate a new `assign_tasks`.

Step 7. If necessary, create the file `laura_vis_data` (see Section 5.12 on page 48 for more detail.)

Step 8. Modify `tdata` file (see Section 5.5 on page 36) to define the gas model condition for your specific simulation. The species order in this file must match the species order used in the prior version of LAURA. The other data files should not be changed.[A21]

Step 9. Run LAURA as usual—see Section 4 on page 10.

[A21] These files may be changed if different thermodynamic model, curve-fit data, or thermochemical reaction is needed—see Section 5.5 on page 36 for more detail.

Appendix B

Additional Molecular Band Systems

This appendix lists molecular band systems available in addition to those listed in Section 5.6 on page 38. The band systems listed here are generally weak emitters and absorbers, and are therefore not engaged as a default (unlike those listed in Section 5.6 on page 38). Therefore, for these band systems to be engaged, the following binary flags (on=1/off=0) must be present and set equal to 1 in the `hara_namelist_data` file.

`treat_band_c2_br`

A binary flag activating the C_2 Ballik-Ramsay band system.

`treat_band_c2_da`

A binary flag activating the C_2 Deslandres-d'Azambuja band system.

`treat_band_c2_fh`

A binary flag activating the C_2 Fox-Herzberg band system.

`treat_band_c2_mulliken`

A binary flag activating the C_2 Mulliken band system.

`treat_band_c2_philip`

A binary flag activating the C_2 Philips band system.

`treat_band_co3p`

A binary flag activating the CO 3+ band system.

`treat_band_co_angstrom`

A binary flag activating the CO angstrom band system.

`treat_band_co_asundi`

A binary flag activating the CO Asundi band system.

`treat_band_co_triplet`

A binary flag activating the CO triplet band system.

`treat_band_co2`

A binary flag activating the CO_2 band system. A value of 2 activates an approximate nonequilibrium model for UV emission, while a value of 1 assumes Boltzmann emission.

`treat_band_n2_cy`

A binary flag activating the N_2 Carrol-Yoshino band system.

`treat_band_n2_wj`

A binary flag activating the N_2 Worley-Jenkins band system.

`treat_band_n2_worley`

A binary flag activating the N_2 Worley band system.

`treat_band_no_gamma`

A binary flag activating the NO gamma band system.

`treat_band_no_betap`

A binary flag activating the NO beta-prime band system.

`treat_band_no_gammap`

A binary flag activating the NO gamma-prime band system.

`treat_band_o2_sr`

A binary flag activating the O_2 Schumann-Runge band system.

`treat_[?]_photo`

A binary flag activating the molecular photo-ionization mechanism [23] for `[?]` specie, where `[?]` can be o2, n2, and or no. This mechanism is not technically a molecular band system.

Appendix C

Trouble Shooting

This appendix gives some tips if you experience trouble installing or running LAURA. This is far from all-inclusive; it is only a modest attempt to capture some of the experience shared by customers over the years through the community LAURA-users@lists.nasa.gov and private LAURA-support@lists.nasa.gov email lists.

Of course the first step is to be sure you have the latest version of LAURA—releases are announced on the LAURA-news@lists.nasa.gov mailing list.

C.1 Installation

C.1.1 Unterminated Constant / Line Truncated

An error during compilation like,

```
ifort [...] -DDATADIR=\"[some really long path]\" [...] \
  -c -o datadir_file_manager.o datadir_file_manager.F90
In file datadir_file_manager.F90:30
   search_paths(2) = "[some really long pa
                     1
Error: Unterminated character constant beginning at (1)
 In file datadir_file_manager.F90:30
pa
 1
Warning: Line truncated at (1)
make[5]: *** [datadir_file_manager.o] Error 1
```

are due to your installation path violating Fortran's "free-form" line limit of 132 characters. Many compilers do not enforce this, but for those that do, compiler options are usually available to circumvent this problem—for example, gfortran's `-ffree-line-length-none` or g95's `-ffree-line-length-huge`.

C.2 Running

LAURA is designed to recognize when obvious mistakes in input are made: these errors will be reported to you via standard output (nominally, the screen). Even when the code runs successfully, the user should always check the standard output for warning messages.

The ideas given below are mostly for when the code blows up, and the code's output "advice" is not very helpful (NaN detected, segmentation faults, etc.).

Most of the time, problems running the code can be traced to the use of an inadequate grid, block boundary condition errors, or other configuration missteps. The grid should look "nice". There should be "gentle" stretching factors (shoot for

less than 1.15), the grid should be as orthogonal as possible—especially in viscous regions, and dramatic changes in distance or orientation from one grid line to the next should be avoided. Furthermore, the grid minimum spacing (at walls) should be appropriate to the problem you are running. In other words, "viscous" grids should have reasonably low cell-Reynolds numbers at the wall (`re_cell` = 0.1–1.0 depending on the amount of diffusion); and one should not trying to run inviscid flow on a "viscous" grid.

The first thing to try after a case blows up is to raise the inviscid and viscous relaxation factors—see Section 5.4.10 on page 27. Typically, you can start with values like `rf_inv=4` and `rf_vis=3` and end with values like 3 and 1.5, but sometimes you need to raise them much higher (e.g., 20 and 10 or 200 and 100) to get a solution past a particularly nasty transient. Other things to try include running first-order, increasing the frequency of Jacobian, transport, and inter-processor communication updates, or successively ramping up the Mach number.

Running on a coarser version of the grid sometimes helps get the solution going–see the `coarsen` utility described in Section 7.2 on page 60.

C.2.1 NaNs

NaNs (Not a Numbers) are a type of IEEE Floating Point Exception. Most compilers have options (either during compilation or at runtime via environment variables) to trap these exceptions and stop the code in its tracks when one occurs. For example, the g95 compiler has environment variables of the form `G95_FPU_INVALID` that can control this while the Intel compiler has the `-fpe` compilation option. Note: to pin point where in the code the exception occurs, you will also want to turn on tracing (e.g., `-traceback` for Intel and `-ftrace=full` for g95) and symbols (`-g`) so the compiler will give you the precise source-code line number.

C.2.2 Segmentation Faults

Segmentation faults usually mean you've hit a shell-based memory limit, you've run out of memory, or you've uncovered a coding error.

To check the first, remove your shell memory limits, i.e.,

```
csh: limit stacksize unlimited
bsh: ulimit -s unlimited
     ulimit -d unlimited
     ulimit -m unlimited
```

and try re-running. Note: For an MPI job, these have to be in your shell startup environment (e.g., `.cshrc` or `.bashrc`).

To check the second, try a smaller case, request more resources (e.g., use less cores per node), and/or use the batch option of the `top` command (`-b`) to monitor memory usage.

For the third, please see Support, section D on the next page.

Appendix D

Support

Because LAURA's primary purpose is to serve NASA missions and it is not offered as commercial software, support is available on an "as time allows basis" via two channels: the public LAURA-users@lists.nasa.gov community email list, which you are encouraged to join via lists.nasa.gov/mailman/listinfo/LAURA-users, and the private LAURA-support@lists.nasa.gov email list.

If your issue is not proprietary or otherwise sensitive, you are strongly encouraged to use and become a member of the LAURA-users@lists.nasa.gov list.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-08-2010	Technical Memorandum	

4. TITLE AND SUBTITLE
LAURA Users Manual: 5.3-48528

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Alireza Mazaheri, Peter A. Gnoffo, Christopher O. Johnston, and Bil Kleb

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
432938.11.01.07.43.05.01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-0001

8. PERFORMING ORGANIZATION REPORT NUMBER
L–19910

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSOR/MONITOR'S ACRONYM(S)
NASA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
NASA/TM–2010–216836

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified-Unlimited
Subject Category 64
Availability: NASA CASI (443) 757-5802

13. SUPPLEMENTARY NOTES
An electronic version can be found at http://ntrs.nasa.gov.

14. ABSTRACT
This users manual provides in-depth information concerning installation and execution of LAURA, version 5. LAURA is a structured, multi-block, computational aerothermodynamic simulation code. Version 5 represents a major refactoring of the original Fortran 77 LAURA code toward a modular structure afforded by Fortran 95. The refactoring improved usability and maintainability by eliminating the requirement for problem-dependent re-compilations, providing more intuitive distribution of functionality, and simplifying interfaces required for multi-physics coupling. As a result, LAURA now shares gas-physics modules, MPI modules, and other low-level modules with the FUN3D unstructured-grid code. In addition to internal refactoring, several new features and capabilities have been added, e.g., a GNU-standard installation process, parallel load balancing, automatic trajectory point sequencing, free-energy minimization, and coupled ablation and flowfield radiation.

15. SUBJECT TERMS
Aerodynamics; Aerothermodynamics; Ablation; Radiation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email: help@sti.nasa.gov)
U	U	U	UU	88	19b. TELEPHONE NUMBER *(Include area code)* (443) 757-5802