NASA/TM—2014-216623

Using the NASA GRC Sectored-One-Dimensional Combustor Simulation

Daniel E. Paxson and Vishal R. Mehta
Glenn Research Center, Cleveland, Ohio

March 2014

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 Reports Server, thus providing one of the largest collections 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 has 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 cosponsored 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, 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 to *help@sti.nasa.gov*

- Fax your question to the NASA STI Information Desk at 443–757–5803

- Phone the NASA STI Information Desk at 443–757–5802

- Write to:
  STI Information Desk
  NASA Center for AeroSpace Information
  7115 Standard Drive
  Hanover, MD 21076–1320

NASA/TM—2014-216623

Using the NASA GRC Sectored-One-Dimensional Combustor Simulation

Daniel E. Paxson and Vishal R. Mehta
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

March 2014

This report is a formal draft or working paper, intended to solicit comments and ideas from a technical peer group.

This report contains preliminary findings, subject to revision as analysis proceeds.

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA Center for Aerospace Information
7115 Standard Drive
Hanover, MD 21076–1320

National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Available electronically at http://www.sti.nasa.gov

Using the NASA GRC Sectored-One-Dimensional Combustor Simulation

Daniel E. Paxson and Vishal R. Mehta*
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Introduction

The following document is a user manual for the NASA GRC Sectored-One-Dimensional (S-1-D) Combustor Simulation. It consists of three sections. The first is a very brief outline of the mathematical and numerical background of the code along with a description of the non-dimensional variables on which it operates. The second section describes how to run the code and includes an explanation of the input file. The input file contains the parameters necessary to establish an operating point as well as the associated boundary conditions (i.e., how it is fed and terminated) of a geometrically configured combustor. It also describes the code output. The third section describes the configuration process and utilizes a specific example combustor to do so. Configuration consists of geometrically describing the combustor (section lengths, axial locations, and cross sectional areas) and locating the fuel injection point and flame region. Configuration requires modifying the source code and recompiling. As such, an executable utility is included with the code which will guide the requisite modifications and insure that they are done correctly.

1.0 Background

Before proceeding, potential code users should consult the following documents which accompany the software or are available online.

1. Paxson, D.E., "A Sectored-One-Dimensional Model For Simulating Combustion Instabilities In Premix Combustors," AIAA-2000-0313, January 2000, and NASA/TM—1999-209771, January 2000.
2. DeLaat, J.C. and Paxson, D.E.: "Characterization and Simulation of the Thermoacoustic Instability Behavior of an Advanced, Low Emissions Combustor Prototype," prepared for the 44th Joint Propulsion Conference and Exhibit. AIAA-2008-4878, NASA/TM—2008-215291, July 2008.
3. Paxson, D.E., "Simulation of Combustion Instabilities: A Sectored-One-Dimensional Approach," presented at the 2007 Propulsion Controls and Diagnostics Workshop, Cleveland, Ohio

These documents provide a more thorough description of the code in terms of both theory and capability. What follows is a brief summary of both.

1.1 Governing Equations

The simulation models a combustor by dividing it into several segments (up to 5). Each segment has uniform cross sectional area. Within each segment the simulation (a.k.a. code in this document) numerically integrates the following governing differential equations for a calorically perfect gas, written below in non-dimensional (a.k.a. normalized) form

* This work was supported while the author was a NASA USRP intern at GRC during the summer of 2010

$$\frac{\partial \overline{w}}{\partial t} + \frac{\partial \overline{F}(\overline{w})}{\partial x} = \overline{S}(\overline{w}) \tag{1}$$

where

$$\overline{w} = \begin{bmatrix} \rho \\ \rho u \\ \dfrac{p}{\gamma(\gamma-1)} + \dfrac{\rho u^2}{2} \\ \rho z \end{bmatrix} \tag{2}$$

and

$$\overline{F} = \begin{bmatrix} \rho u \\ \dfrac{p}{\gamma} + \rho u^2 \\ u\left(\dfrac{p}{\gamma-1} + \dfrac{\rho u^2}{2}\right) \\ \rho u z \end{bmatrix} \tag{3}$$

The distance, x has been normalized by the overall combustor length, L. The time, t has been normalized by the characteristic wave transit time, L/a^*, where a^* is the speed of sound at a chosen reference state. The pressure, p and density, ρ have been normalized by their respective reference values, p^* and ρ^*. The axial velocity, u has been normalized by a^*. The mass fraction of reactant (fluid containing chemical energy) is z. In this code, the fluid is either reactant or product of combustion. The ratio of specific heats is denoted by γ.

For this non-dimensionalized form of the equations, the equation of state is written:

$$p = \rho T \tag{4}$$

The non-dimensional speed of sound is simply $\sqrt{T}$.

The source vector is:

$$\overline{S}(\overline{w}, x) = \begin{bmatrix} 0 \\ \dfrac{\partial}{\partial x}\left(\dfrac{\varepsilon_t}{\mathrm{Re}^*}\left(\dfrac{\partial u}{\partial x}\right)\right) + s_2 \\ \dfrac{\partial}{\partial x}\left(\dfrac{\varepsilon_t}{\mathrm{Re}^*}\dfrac{\partial}{\partial x}\left(\dfrac{u^2}{2} + \dfrac{T}{(\gamma-1)\,\mathrm{Pr}_t}\right)\right) + q_0 R \\ \dfrac{\partial}{\partial x}\left(\dfrac{\varepsilon_t}{\mathrm{Re}^*\,\mathrm{Sc}_t}\left(\dfrac{\partial z}{\partial x}\right)\right) - R + s_4 \end{bmatrix} \tag{5}$$

It contains contributions from the reaction, and turbulent eddy diffusion. The Reynolds number, Re^* is defined as $\rho^* a^* L/\mu$. The turbulent viscosity ratio, ε_t is defined as the ratio of turbulent (eddy) to

molecular viscosity, μ_t/μ. The term q_0 is the reactant heat of reaction.[1] R is the reaction rate, defined below. Pr_t and Sc_t are the turbulent Prandtl and Schmidt numbers respectively. These parameters are specified by the user in an input file described in a later section. It is noted here however that, due to the simplicity of the governing equations (i.e., 1-D, no turbulence model), and the practical necessity of relatively crude numerical grids, the turbulent viscosity (and associated Reynolds number) do not have particular meaning. The ratio ε_t/Re^* may be thought of as a single diffusion coefficient which a user will input and "tune" to represent a desired feature of the combustor (i.e., a swirl stabilized flame region).

Two other terms appear in the source vector, Equation (5). They are s_2 and s_4. The term s_2 is a user defined 'pink' noise source of momentum that is only implemented in the vicinity of the fuel injector. It is intended to mimic effects of shed vortices, shear layers, etc. that are seen on actual combustors and thought to be responsible for the typical 'noise floor'. The term s_4 is another user defined term representing added fuel. It is only implemented at the fuel injector location. Note that in this code the addition of fuel is assumed to add no mass to the system. This is consistent with the assumption that the mass flow rate of fuel is negligible compared to that of air.

1.2 Combustion Model and Reaction Rate

The combustion (or reaction rate) model, R in Equation (5) has the form:

$$R = K_0 \rho z(\zeta_1 - \zeta_2 z)\begin{Bmatrix} 1-\left(T_{ign}/T_i\right) & ;T_i > T_{ign} \\ 0 & ;T_i < T_{ign} \end{Bmatrix} \tag{6}$$

where K_0 is the reaction rate constant, ζ_1 and ζ_2 are constants defining the type of reaction (assigned respective values of either 1.0, 1.0 or 1.0, 0.0), T_{ign} is the ignition temperature and T_i is the temperature in the i^{th} numerical cell.

1.3 Flameholder

The flameholder is not shown per se in the governing equations. It is manifested as a fixed spatial gradient, occurring over 5 percent of the combustor length, whereby the value of the diffusion coefficient ε_t/Re^* changes from 0 to the maximum value specified by the user in the input file. The location of this transition is also specified by the user.

1.4 Numerical Method

The simulation numerically integrates the above equations of motion using a very simple, second-order MacCormack scheme. Artificial viscosity has been added in order to damp non-physical oscillations in the vicinity of strong gradients such as those brought about by the combustion process. Artificial viscosity, like actual viscosity, has an associated user specified diffusion coefficient. The choice of value is explained in Section 2.2.

1.5 Boundary Conditions

Boundary conditions, representing the termination of the computing domain, may be imposed as either partially opened, fully open, or choked inflow (e.g., constant mass flux) ends. In either case, the code anticipates the flow direction and applies appropriate (e.g., well-posed) states. If outflow is anticipated, only the static pressure is imposed. The remaining information (density, velocity, and

[1] Generally, $q_0 = \dfrac{h_f}{a^{*2}\left(a/f+1\right)}$, where h_f is the fuel lower heating value, and a/f is the air-to-fuel ratio.

reaction fraction) is obtained via characteristic jumps across the adjacent numerical cell. If inflow is anticipated, total pressure and temperature, and reaction fraction are imposed. The remaining unknown quantity (velocity) is obtained through iteration and characteristic jumps across the adjacent numerical cell. Details of the boundary conditions may be found in the reference list at the end of this document. It is noted that the reactant fraction can also be specified at an inlet, allowing the possibility of simulating premixed combustion. This would require the fuel source term, s_4 to be set to 0.

1.6 Crossing Sectors/Area Change

Compatibility between sectors is ensured by combining a fully-open type boundary condition for one sector and a partially-open boundary condition of another. For example, consider a case where the area abruptly enlarges from Sector 1 to Sector 2. The flow is from left to right. A guess is made for the exit pressure of Sector 1. The state of the fully-open image cell for this sector is used to calculate stagnation properties. These, in turn, are used to determine the image cell state for the partially-open configuration of Sector 2. The area of each sector is known. Thus, for the assumed exit pressure of Sector 1, the mass flux (ρuA) in the image cell of each sector is also known. The proper choice of Sector-1 exit pressure then is that which yields the same mass flux in both sectors. This is obtained in the code through an iteration technique.

If the geometry and flow conditions are such that flow is from a large to a small area sector, a similar iteration can be established using the fully-open inflow and partially-open outflow boundary conditions described above.

At present, the area change aspect of the simulation is limited to unchoked flow; however, the method just described can easily be modified to accommodate choked flow with abrupt area change.

Note that the structure of this code is such that the geometry of the particular combustor being simulated is embedded into the code. Modification of the code to accept user defined geometry is discussed in the section entitled Configuration.

Note also that the boundary conditions at either end of the combustor of interest can be changed without code modification. That is to say, they can be changed within the input file described below. This is convenient since instabilities are often tied to a particular boundary condition. This means that a given combustor can often be stabilized by simply altering the boundary condition. This can be helpful for example if the dynamic development of an instability is sought.

2.0 Code Operation

2.1 Compilation

The simulation consists of four FORTRAN 77 source files, and an input file. The main pieces of the code are the subroutines contained in the source file **s1d_demo.f, inter.f,** and **bc_ideal.f**. There is also a main program called **tester_s1d.f**, from which the subroutines are called. In a simulation in which control was being implemented, this program would probably not be used; however, it does provide insight into how the flow of data occurs, and how the subroutines exchange input and output. Stated another way **tester_s1d.f** simply calls the workhorse subroutines appropriately, and provides output for post-processing. The source files must be compiled and linked to create an executable program.

Examination of **tester_s1d.f** will show a four element array called ZZP. This is the array by which the main program passes information to the subroutines. The significance of each element of the array is explained in the source code. As delivered, this array is used to obtain pressure at two locations, as well as to introduce noise into the simulation (s_2) and to set the fuel flow rate variations (s_4). However, although the fuel variation parameter exists, the lines which supply the variation have been commented out so that, for this demonstration code, the fuel flow rate is constant. The two pressure sensor locations (ZZP(3) and ZZP(4)) are specified by the user in source code **s1d_demo.f**. Changing them is simple, but requires recompilation of the code.

2.2 Input File

The input file for the simulation is called **flametub.dat**. Relevant portions of the file are shown below. Note that the italicized text shown does not actually appear in the input file. It has been added here for descriptive purposes. All input file quantities are non-dimensional. For the particular simulation provided, the reference conditions are described below the input file. This is the information required to convert non-dimensional units into physical units. For the most part, all of the information is specific to the combustor being simulated, and the particular operating point.

```
  201          Number of numerical cells plus 1 (cell #1 is first interior, cell #200 is last interior, 0 and 201 are image cells)
  1.30000      Ratio of specific heats
  0.00000      friction coefficient.
  0.00150      The ratio εt /Re* in Equation (5)
  1.00000      Turbulent Schmidt Number
  1.00000      Turbulent Prandtl Number
  1.00000      ζ1 in Equation (6)
  0.00000      ζ2 in Equation (6)
  4.20000      q0 in Equation (5)
 25.00000      K0 in Equation (6)
  1.30000      Tign in Equation (6)
  0.71534      Static pressure at outlet of the combustor.
  2.30000      Temperature at outlet of the combustor.  Only used if inflow occurs.  It is assumed as a stagnation value
  0.00000      Reactant fraction at outlet of the combustor. Only used if inflow occurs.
  0.10500      Open area of right end of combustor divided by nominal cross section (see * below)
  1.00500      Total Pressure at left end of combustor for open end b.c.  Mass flux for constant flux b.c.
  1.00000      Total Temperature at left end of combustor.
  0.00000      Reactant fraction at left end of combustor.  If >0, allows for simulation of premix scenarios
  0.17000      Open area at left end of combustor divided by nominal cross section (see § below)
  0.00250      Numerical time step
  0.25000      Constant for artificial viscosity.  May be lowered until 'wiggles' are small near large gradients
  97           Index locating flameholder: INT(xfh/Δx + .5). Must be INT(.03/Δx) cells from any area transition
 35.80000      Reactant mean flow rate if fuel is injected other than at the inlet
  1.004537066970680   1.004543865005031   0.006694292226215   0.000000000000000
  1.004643183514743   1.004650894137265   0.006710413297609   0.000000000000000
  1.004816590889842   1.004789958301161   0.006667714413936   0.000000000000000
  1.004937914671311   1.004883655163755   0.006648896964835   0.000000000000000
  1.005050122467696   1.004973025869399   0.006616015361772   0.000000000000000
  1.005138986212765   1.005040147473687   0.006577026660130   0.000000000000000
      pressure            density             velocity            reaction fraction, z
```

There will be as many rows as there are interior numerical cells plus two. These specify the initial state for the simulation.

.
.

§For the number denoting the open area of the left end, if the value is less than 1.0, the end is partially open. If value equals 1.0, then the end is fully open. If the value is greater than 1.0, then choked inflow is assumed and the end pressure variable in the input file is read as a constant mass flux (i.e., ρu = constant). For the right end, only the fully and partially open options are allowed.

2.2.1 Conversion to Dimensional Units

Table I shows conditions pertain to the supplied simulation.

TABLE I.—CONDITIONS PERTAIN TO THE SUPPLIED SIMULATION

p^*, psia	T^*, R	ρ^*, lbm/ft^3	γ	L, ft	a^*, ft/s
250	1460	0.460	1.30	5.45	1809

All quantities in the simulation are non-dimensionalized by these reference conditions or combinations thereof. Denoting the dimensional quantities with primes, the following relations apply.

Time: $t' = \frac{tL}{a^*}$ sec; Frequency: $f' = \frac{fa^*}{L}$; Hz. $p' = pp^*$ psia; $T' = TT^*$ R; $\rho' = \rho\rho^*$ lb_m/ft^3; $u' = ua^*$ ft/s

2.3 Output File(s)

There are two output files produced by the simulation, **out.dat** and **flametub.new**.

2.3.1 out.dat

This file has two possible formats depending on the selection made at run time. One format contains the states (pressure, density, velocity, reactant fraction) of all the numerical cells at discrete time intervals. The simulation prompts the user for a total simulation time, τ_{sim} and a time specifying the start of output, τ_{out}. An output interval is then computed such that the difference between τ_{sim} and τ_{out} is divided into approximately 500 equal time segments, e.g.,

$$N = INT\left(\frac{\tau_{sim} - \tau_{out}}{500\Delta\tau}\right) \tag{7}$$

where $\Delta\tau$ is the numerical time step specified by the user in the input file. Beginning at τ_{out} the simulation then writes the state of every numerical cell each N time steps. The form of the output is:

```
20
   999  999  99  500
   999  999  99  500
   999  999  99  500
   999  999  99  500
   999  999  99  500
   .
   .
   .
40
   999  999  99  500
   999  999  99  500
   999  999  99  500
   999  999  99  500
   999  999  99  500
```

The numbers, 20, 40, etc. represent the times at which the data was output. Like all of the quantities in this file, they are integers calculated via the expression I=INT(quantity X 1000.0 + .5). The columns which follow the times represent (from left to right) pressure, density, velocity, and reactant fraction respectively. There are PNTS + 1 rows of data following each time stamp, where PNTS represents the number of interior numerical cells plus one, as specified in the first line of the input file described in Section 2.2.

The second output format option consists of the time, and pressure at two locations in the combustor (see Sec. 3.1.1). These are output at a non-dimensional time interval which, for the present version corresponds to 5000 Hz. It may be changed via reconfiguration as described in Section 3.1.1.

The format for the pressure values is floating point, but the numbers are still non-dimensional. The default locations to which the output data correspond are shown in Figure 1. MATLAB scripts accompany the source code, and demonstrate the processing of this information (assuming that the user is familiar with, and has access to MATLAB). Section 2.4 shows sample output.

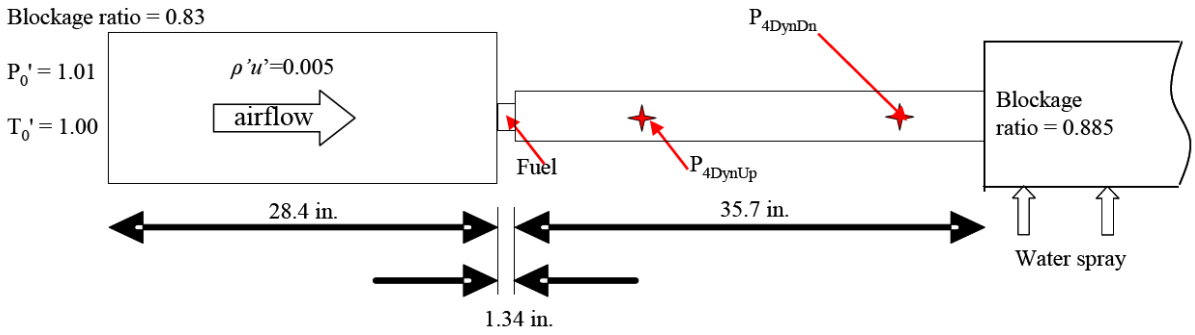


Figure 1.—Supplied Combustor Simulation Configuration

2.3.2 flametub.new

This file is a restart file. It contains all of the parameter values of **flametub.dat** and the current state in all of the numerical cells at time τ_{sim}. Thus, if **flametub.new** is renamed **flametub.dat**, the simulation can pick up where it left off at the end of the previous simulation.

2.4 Execution

When the code is executed, the following output will appear on the display:

```
INITIALIZING.....
ENTER SIMULATION TIME:
```

Recall that the simulation time entered is non-dimensional. One second of real time is 331.91 non-dimensional units for the simulation supplied. This supplied simulation has an instability that is approximately 500 Hz (a period of 0.664 non-dimensional time units). Thus, a simulation of only about 20 time units will cover numerous cycles.

After entering a total simulation time (and hitting Enter), the following prompt appears:

```
ENTER OUTPUT START TIME:
```

This is the non-dimensional time at which output commences.

Next, the code gives the following prompt:

```
ENTER PERTURBATION LEVEL
```

This refers to a random noise level that is introduced into the simulation (see Section 1.0, "Background" that describes Equation (5)). It is introduced near the fuel injection point in the form of fluctuations in momentum (the s_2 term in Eq. (5)). The rationale is that much of the combustor noise emanates from vortex-type shedding and shear flows. The noise is not truly random. It is filtered such that there is very little content at frequencies above 5,000 Hz (for this simulation). This is consistent with observations. The perturbation level can be set to whatever the code will sustain without becoming unstable. Numbers above 0.85 are probably unreasonable. For simulations of several experiments, a value of 0.5 has yielded good results. Changing the frequency range of this noise requires modifying the source code and is therefore detailed in the configuration section of this manual.

It is worth pointing out that this noise can sometimes interfere with the thermoacoustic coupling of weak instabilities. However, it can also provide acoustic energy to initiate instabilities. As such, when attempting to simulate actual experimental rigs, it is worthwhile to parametrically vary this term and compare results with data in terms of signal to noise.

Finally, the following prompt will appear:

```
ENTER OUTPUT TYPE 1=PSD 2=PVWAVE
```

This refers to the two different output types described in Section 2.3, “Output File(s).” Option 1 gives pressure at two locations in the combustor. Option 2 gives information throughout the combustor.

After choosing an output type, hit Enter. The code will commence simulation with the message

```
SIMULATING.....
```

When the specified simulation time is reached, the following message will appear

```
WRITING CURRENT STATE.....
```

Output may then be processed. Example MATLAB scripts for doing so are included with the source code. They are named **spectrum.m** for type 1 output and **wave.m** for type 2 output. Plots from each of the scripts are shown in Figure 2 and Figure 3 for the supplied configuration and input file.

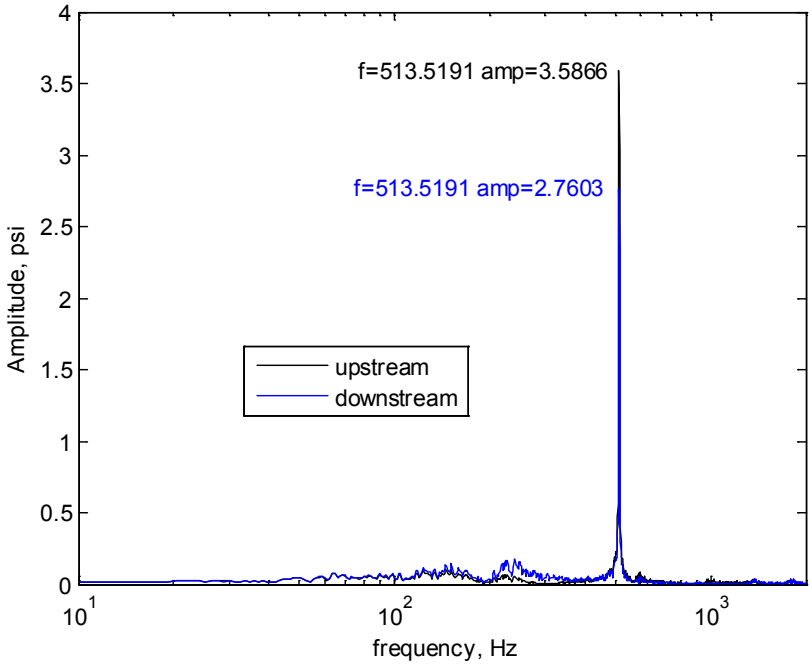


Figure 2.—Sample plot from the spectrum.m data processing script showing instability amplitude at two locations in the sample combustor simulation described by Figure 1. The 663 units of simulation time (2.0 sec) were used, with a perturbation level of 0.5.

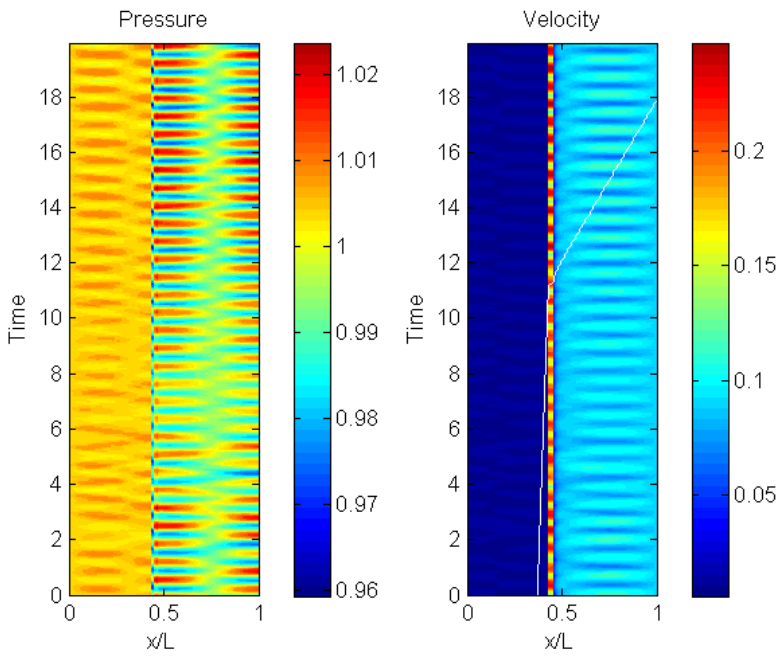


Figure 3.—Sample plot from the wave.m data processing script showing contours of pressure and velocity over 20 units of simulation time with a perturbation level of 0.5. A particle pathline is also shown on the right. The configuration is that of Figure 1

3.0 Configuration

In order to simulate a particular combustor, it is necessary to modify the source code files since they ultimately contain the geometric description of the combustor (sector lengths, sector cross-sectional areas, injector locations, etc.). This section will guide a user through a manual configuration process and it will introduce a computer program that configures the simulation semi-automatically, after prompting for several inputs specific to the combustor under consideration.

3.1 Manual Configuration

As mentioned in Section 2.1, the combustor simulation is a collection of four source code files and one input file. The following three source files must be modified in order to describe a specific combustor: **tester_s1d.f** , **s1d_demo.f**, and **inter.f,** and. Before beginning the configuration process, it is recommended that all files provided are copied and placed in a separate directory.

3.1.1 tester_s1d.f

In **tester_s1d.f**, the only configuration necessary is the non-dimensional data output frequency, *fsamp*. In order to set this value, open the file and locate the line commented with "c FirstConfig." The next line is *fsamp*, which can be calculated by Equation (8).

$$fsamp = \frac{f' L}{a^*} \tag{8}$$

where f' is the data output frequency in Hz, L is the combustor length, and a^* is the reference speed of sound. Below is a sample of the source code modified for a 5000 Hz data output frequency, and for a combustor 65.44 in. long with a speed of sound of 1809 ft/s. (note combustor length is divided by 12.0 to

convert to units consistent with the speed of sound). Per FORTRAN formatting, "fsamp = " should begin on or after the 7th column as shown:

```
c FirstConfig
        fsamp =    15.072700
c End FirstConfig
```

Note that an actual equation for *fsamp* as described above can be coded, or the result of the equation (i.e., a single number) can be used.

3.1.2 s1d_demo.f

In **s1d_demo.f**, the first configuration step involves declaring the indices which locate the sectors. Find the section of code between the comment lines "c FirstConfig" and "c End FirstConfig,":

```
c FirstConfig
      INTEGER INDEX1, INDEX2
c End FirstConfig
```

In this section the indices must be declared as integer type. For a configuration with *n* sectors, there will be *n* – 1 indices, therefore *n* – 1 indices will need to be declared as integer type. For the supplied simulation configuration shown in Figure 1, there are three sectors, and thus two INDEX integers. The entire combustor is divided into *pnts* – 1 numerical cells. Each index represents the cell (axial location) where an area transition takes place.

The format for the declarations should be as shown above with the INTEGER type declaration followed by "INDEX" concatenated with the index number. As stated previously in reference to "fsamp," "INTEGER" should start on or after the 7th column as per FORTRAN formatting (reminder: FORTRAN also has a 72 column line limit).

The next configuration step involves identifying the location of the sector changes and the area ratios between the sectors. In addition, the index of the sector change to the right of the flameholder is identified to serve as a boundary for the distribution of the turbulent diffusion coefficient, ε_t/Re^* (see Sec. 1.3). The index value for a sector change is calculated as follows.

$$INDEX = INT\left(\frac{total\ \#\ of\ cells}{combustor\ length} * location + 0.5\right) \tag{9}$$

where *location* is the distance from the left (inlet) end of the simulation. For example, considering the supplied simulation, the overall length is 65.44 in. The leftmost sector ends at *location*=28.4 in. (see Figure 1). There are 200 numerical cells for the simulation. Thus, *INDEX*2 = *INT*(200 / 65.44 * 28.4 + 0.5) = 87. Proceed through **s1d_demo.f** to the line starting with "c SecondConfig" and replace the existing indices with the location indices of the sector changes of the desired combustor configuration while adding any additional indices. Keep in mind that indexing proceeds from right to left whereby the rightmost sector change corresponds to "INDEX1." Below is the section of code where the modification must occur:

```
c SecondConfig
        Index1 =  91
        Index2 =  87
c                AREA RATIOS
        AR1 =  0.208125
        AR2 =  0.029444
c End SecondConfig
```

In addition to the indices, the sector cross-sectional area ratios have to be specified. It is important to note that sector changes correspond to either an expansion or a contraction (with flow presumed from left to right). In either case the area ratio is specified as "smaller over larger." That is to say, area ratios are always less than 1.

It is assumed in the code that turbulent diffusion is only significant in sector where combustion occurs. As such, the turbulent viscosity ratio distribution is prescribed such that it is zero outside of the sector where the flameholder is located. Within the combustion sector, it rises rapidly (over 5 percent of the combustor length) from zero to its maximum prescribed value at the flameholder index. From there to the end of the sector it gradually diminishes to zero. The numerical cell at which the turbulent viscosity ratio has returned to zero is stored in the variable RINDEX. If the flameholder is located in a sector other than Sector 1 (the rightmost sector), then RINDEX must be assigned a value corresponding to the right hand end of the sector where the flameholder is located. This assignment should immediately precede the "`c End SecondConfig`" line above and take the form:

```
        AR2 =  0.029444
         RINDEX = XX
c End SecondConfig
```

The third configuration of **s1d_demo.f** involves setting filter parameters for the noise. Though the simulated noise is random, it is not truly white. Sources of combustor noise are typically in a frequency range. In order to reflect this reality it is necessary to employ a low-pass and a high-pass cutoff frequency, essentially creating a bandpass filter. The section to be modified to filter the noise in this manner is shown below:

```
c ThirdConfig
        lpcnt =    26
        df1 =  0.940300
        cf0 =  0.970100
        cf1 = -0.970100
c End ThirdConfig
```

The variable *lpcnt* essentially achieves the low pass filtering process by determining the number of simulation time steps between calls to a random number generator. It can be determined by Equation (10)

$$lpcnt = INT\left(\frac{1.0}{f_{cutoff}\, dti}\right) \tag{10}$$

where f_{cutoff} is the low-pass cutoff frequency and *dti* is the numerical time step. Both parameters are non-dimensional.

The *butter* command in MATLAB can be used to obtain the high-pass, first–order Butterworth filter parameters *cf*0, *cf*1, and *df*1. To do so input the following command into MATLAB:

$$[c, d] = butter\ (1,\ 2.0 * lpcnt * dti * f_{hp},\ 'high')$$

Here f_{hp} is the desired non-dimensional high-pass frequency (for the included example combustor, the dimensional high pass filter frequency is 50 Hz). MATLAB will output vectors *c* and *d*, where the first element of vector *c* is *cf*0, the second element of vector *c* is *cf*1, and the second element of vector *d* is *df*1. The value to be input into the simulation code for *df*1 should be the negative of the *df*1 value MATLAB provides. A MATLAB script, **filterConfig.m**, is supplied with the source code to calculate these noise filter parameters as well as *fsamp* and *lpcnt* based on user-provided combustor length, dimensional data output frequency, cutoff frequency, speed of sound, and time step.

The fourth configuration is responsible for calling either the CONTRACT or the EXPAND subroutines. The section to be modified appears as follows in the source file (for the included simulation with three sectors):

```
c FourthConfig
      CALL EXPAND(AR1, INDEX1, 1)
      CALL CONTRACT(AR2, INDEX2, 2)
c End FourthConfig
```

Based on whether flow traveling from left to right experiences a contraction or an expansion at a sector change, either the CONTRACT or the EXPAND subroutine is called. The format of the call for each INDEX is:

CALL EXPAND/CONTRACT(AreaRatio, Index, ID)

and should be listed for each sector change starting from the rightmost sector change with ID starting as one, incrementing by one for each additional sector change.

The fifth configuration is an excellent reason to use the supplementary computer program to automatically configure the simulation source code (described in Sec. 3.2). Here the code performs a series of numerical cell 'swaps' where first and last interior cells of a sector are temporarily used as image or ghost cells for adjacent sectors. The configuration begins at the following section of the source code:

```
c
c FifthConfig
      PBCN = P(INDEX1)
      UBCN = U(INDEX1)
.
.
.
```

and continues to:

```
      FLUX(1, INDEX1) = FLUX1BCN
      FLUX(2, INDEX1) = FLUX2BCN
      FLUX(3, INDEX1) = FLUX3BCN
      FLUX(4, INDEX1) = FLUX4BCN
```

For a simulation with n sectors, the above block of code needs to be repeated $n - 2$ times. Each time it is repeated all INDEX variables are incremented by 1 (INDEX1 becomes INDEX2, INDEX2 becomes INDEX3). The integer variable ID, which appears approximately 30 lines down from the beginning of the section must also be incremented by 1. This block of program includes DO loops. As per FORTRAN language rules, all DO loops must be assigned new line numbers. In the supplied example combustor simulation, there are three sectors so there is only one occurrence of this code section.

Following this code segment is additional source code that begins with

```
      PBCN = P(INDEXq)
      UBCN = U(INDEXq)
      TBCN = T(INDEXq)
```

And ends with

```
      FLUX(1, INDEXq) = FLUX1BCN
      FLUX(2, INDEXq) = FLUX2BCN
      FLUX(3, INDEXq) = FLUX3BCN
      FLUX(4, INDEXq) = FLUX4BCN
```

The bold **q** appearing at the end of the INDEX variable in the above lines should be an integer of value $n - 1$. The value of ID in this segment of code should also be $n - 1$.

Due to the relatively time consuming (and error prone) process of altering the indices as described above, a supplementary computer program described in Section 3.2 may prove useful, as it somewhat automates this process.

The final change in the fifth configuration step of **s1d_demo.f** involves replacing the dynamic pressure indices based on where the upstream and downstream dynamic pressures are to be recorded. The source code to be changed is:

```
      zzp(3) = p(117)
      zzp(4) = p(176)
c End FifthConfig
c
      RETURN
      END
```

where "117" and "176" should be replaced with the indices corresponding to the dynamic pressure locations of interest.

3.1.3 inter.f

The configuration of **inter.f** involves specifying where the noise is introduced and where the fuel injector is located. Proceeding to the first configuration starting at "c FirstConfig," the location where noise is to be introduced and the width of the noise distribution need to be specified. The code to be modified is:

```
c FirstConfig
        IF (I .GE.  88 .AND. I .LE.  90 ) THEN
c End FirstConfig
```

The noise location for this specific example is between numerical cells 88 and 90. The noise can be added anywhere; however, typically it is added near the fuel injector since injectors are a large source of noise.

The next modification is specifying the location of the fuel injection. Proceeding to "c SecondConfig"

```
C       FUEL INJECTION
c SecondConfig
        IF (I .EQ.  89 ) S(4, I) = S(4, I) + LSSC
c End SecondConfig
```

replace "89" with the cell index corresponding to the location of fuel injection.

The third and fourth configurations are exactly the same as the first and second configurations and follow "c ThirdConfig" and "c FourthConfig" in the code, respectively. The source code **inter.f** is where the interior numerical integration is performed. It is a two-step numerical scheme and thus requires the noise and fuel injection locations to be specified twice.

At this point, manual configuration of the program has been completed. To run the program, the three modified source code files along with the fourth unmodified file **bc_ideal.f**, have to be compiled, and linked.

3.2 Configuration Utility

There exists a simpler method to configure the simulation. A program has been developed that takes the parameters required and configures the source code based on those parameters. This configuration utility can save a significant amount of time in the configuration process and an option exists within the program to recognize and prevent common configuration errors.

The configuration utility is written in FORTRAN 90. In order to run the configuration utility, first compile the FORTRAN source code file **S1D_CONFIGURATION.f.**[2] The configuration utility is valid for a maximum of five sectors. The program needs to be located in the same directory as the four source code files. Maintain a copy of all of the original source files in a separate directory. When the configuration utility is run, two options are provided: an option for an operator who is familiar with the program and an option for an operator who requires guidance.

3.2.1 Familiar Operator

If the first option is selected, the following parameters should be known ahead of time and input when prompted:

- Non-dimensional Data Output Frequency
- Number of Sectors
- Cell indices of Sector Changes
- Area Ratios between sectors or Sector Areas or Sector Diameters
    - If the area ratio option is chosen, an expansion or contraction must be specified based on flow moving from left to right
- Flameholder Cell Index
- Fuel Injector Cell Index
- Noise Cell Index and Noise Distribution Width
- Low-pass Filter Parameter, *lpcnt*
- Butterworth Filter Parameters ($cf0$, $cf1$, $df1$)
- Upstream Dynamic Pressure Cell Index
- Downstream Dynamic Pressure Cell Index

3.2.2 Guided Configuration

The second option is for operators who are not extremely familiar with the simulation or who would like to be cautious with the simulation configuration process. The second option allows for common mistakes (such as incorrect direction of indexing) to be identified and marked as an error, prompting a reentry of the parameter throwing the error. The following parameters are requested:

- Non-dimensional Data Output Frequency, or:
    - Dimensional Data Output Frequency, and
    - Combustor Length, and
    - Speed of Sound (same units as combustor length)
- Number of Sectors
- Cell Indices of Sector Changes, or:

[2] There is a FORTRAN 77 version that also comes with the code called **S1D_CONFIGURATION_77.f**. This has been tested, but not as extensively as the FORTRAN 90 version.

    - Number of Numerical Cells, and
    - Dimensional Location of Sector Change
- Area Ratios Between Sectors / Sector Areas / Sector Diameters
    - If the area ratio option is chosen, an expansion or contraction must be specified based on flow moving from left to right
- Flameholder Cell Index
- Fuel Injector Cell Index
- Noise Cell Index and Noise Distribution Width
- Low-pass Filter Parameter, lpcnt, or:
    - Time Step, and
    - High- and Low-Pass Cutoff Frequencies, and
    - Dimensional Data Output Frequency (if not provided previously)
- Butterworth Filter Parameters ($cf0$, $cf1$, $df1$)
- Upstream Dynamic Pressure Index
- Downstream Dynamic Pressure Index

Upon completion, the configuration utility will rewrite the three source code files mentioned previously. The only step left is to compile and link the four source code files and then run the simulation.

The following problems are known to throw errors: invalid number of sectors (i.e., greater than five sectors), indexing proceeding from left to right, locations exceeding combustor length, indices exceeding number of numerical cells, flameholder preceding fuel injector, and upstream dynamic pressure index greater than the downstream dynamic pressure index.

If compilation errors or simulation runtime errors occur, verify that inputs are correct and rerun the configuration utility. If the errors still occur, proceed with the manual configuration described in the body of this document.

Appendix

In this section of the manual, some pointers are provided that can be helpful when a simulation is "started from scratch." In other words, if a given combustor has been configured, a user must provide an initial state for every computational cell of which the simulation is comprised. A good approach for guessing such states would be to suppose that it is uniform, at zero velocity, and at pressure and temperature equal to the inlet stagnation value. This appendix is designed to get the user from this state to a legitimate operating point with combustion. A generic combustor configuration will be used to illustrate the process. Details of this configuration will not be presented since they are not particularly germane.

A.1 Establishing Flow

The first step is to establish cold premixed flow within the combustor. Populate the states of the input file (see Sec. 2.2) with uniform pressure, temperature, and velocity. The value of the reactant fraction should be zero. Set all the other parameters of the input file to values appropriate for an operating point, with two exceptions.

1. The reactant mean flow rate (non-dimensional fuel flow rate) should be set to zero.
2. The inlet reactant fraction should be set to 1.0. This amounts to introducing a cold premixed combustible mixture into the simulation through the inlet.

Run the simulation for a significant amount of time in order to establish steady flow. Here, "significant" means on the order of the distance from inlet and injector divided by the estimated mean velocity. Care must be exercised however, as reactant (i.e., fluid with z>0.0) should only be present up to and including the sector where combustion is supposed to occur. It is not generally necessary to completely fill the reacting sector with reactant. Filling to some point that includes the flameholder region usually suffices. When this scenario has been achieved save the restart file *flametub.new* as *flametub.dat*, overwriting the original *flametub.dat* file.

A.2 Establishing a Flame

The second step is to establish a flame with the premixed inlet flow. In order to do so, the initial conditions at a sufficient number of numerical cells must be altered to initiate a reaction. Specifically, cells need to be manually set to an approximation of their post-reaction state (high T, z = 0). From the non-dimensional equation of state, Equation (4), where p is the non-dimensional pressure, ρ is the non-dimensional density, and T is the non-dimensional temperature, it can be seen that a reduction in density corresponds to an increase in temperature.

By adequately lowering the density of a sufficient number of numerical cells near the flameholder, the temperature corresponding to complete combustion can be established. The recommended density to enable ignition and sustained combustion can be calculated as follows.

$$\rho = \frac{p}{1+(\gamma-1)q_0} \tag{11}$$

Typically, setting approximately 3 percent of the total number of numerical cells in the vicinity of the flameholder to these conditions will be sufficient for ignition to occur; however, ignition and sustained combustion are sensitive to a variety of other parameters and therefore require some iteration to determine the appropriate number of numerical cells requiring ignition states. The numerical cells upstream of the cells set for ignition should have their reactant fraction set to 1.0; all other cells should have reactant fractions set to 0.0. It is important to reflect this in the boundary conditions by setting the "reactant fraction at the right end of the combustor" to 0.0. Fuel injection should remain off (reactant mean flow

rate set to 0.0). Letting the simulation run for a while will establish a stable flame and reaction profile in the combustor. Figure 4 shows the corresponding temperature distribution with the hot products progressing downstream. Note that the flame remains at the same location and does not migrate downstream. If the diffusion coefficient is too low, the flame front will migrate downstream simulating blowout or blowoff.

A.3 Transitioning to Fuel Injection

Once a flame is established for the premixed inlet flow, save the restart file *flametub.new* as *flametub.dat*. Next lower the reactant fraction at the left end of the combustor gradually (for example, try changing it from 1.0 to 0.9) while running the program for a period similar to that described in Section A.1. Figure 5 shows the convection of the reduced inlet reaction fraction across the combustor to the flamefront. When run for a sufficient amount of time, the final result should be a reduced temperature in the combustion section.

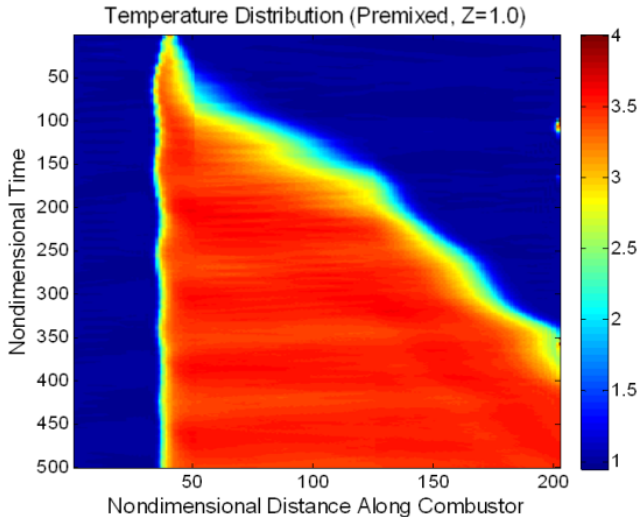


Figure 4.—Temperature contour in the combustor showing the establishment of a flame. Note that time proceeds from top to bottom.

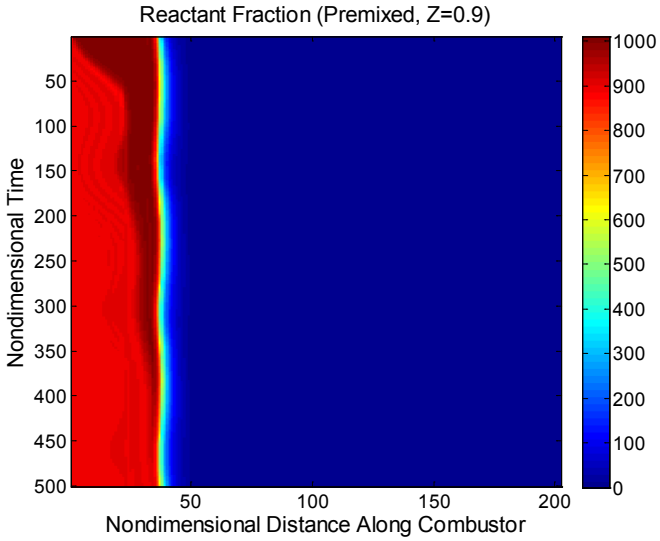
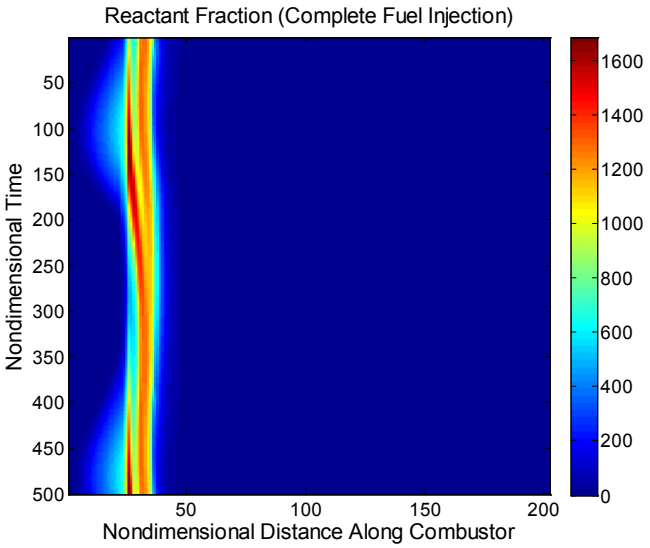


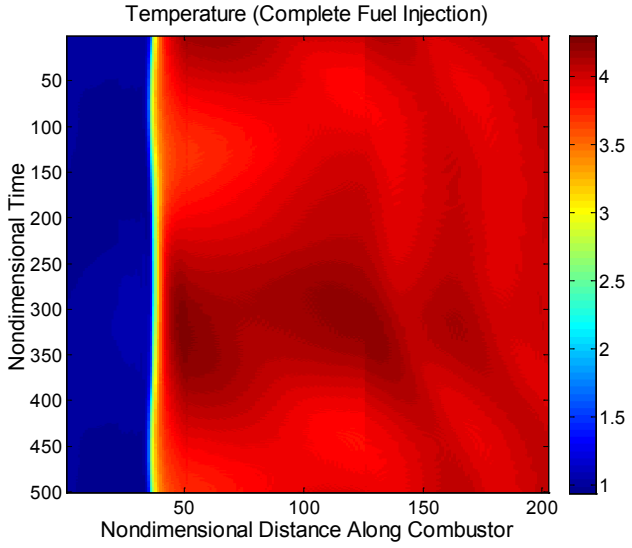
Figure 5.—Contour of reactant fraction (X1000) after reduction from 1.0 to 0.9 at the inlet.

This process should be repeated with ever smaller values of inlet reactant fraction until the flame is on the verge of blowout (i.e., leaving the flameholding region and migrating downstream). When this occurs, save the restart file *flametub.new* as *flametub.dat*.

At this point, the simulation may be restarted with a non-zero value of the reactant mean flow rate parameter. The temperature will rise in the combustion region. When the combustor has reached a steady state (or a limit cycle if it is exhibiting an instability), save the restart file. Repeat this process (incrementally increasing the reactant mean flow rate) until the combustor temperature is near to the expected operating temperature. Save the restart file. The final process is then to incrementally increase the reactant mean flow rate and decrease the inlet reactant fraction until the inlet reactant fraction is zero (no premix) and the combustor temperature is at the appropriate value. Figure 6 shows the combustor in its final state with unstable operation beginning to show (note the periodic oscillations of reactant fraction and temperature).



(a)



(b)

Figure 6.—Contours of (a) reactant fraction (X1000) and (b) temperature with full fuel injection.

Bibliography

Note: These references appear unrelated; however, they describe the boundary condition and reaction models well.

1. Paxson, D. E., "A General Numerical Model for Wave Rotor Analysis," NASA TM 105740, July 1992.
2. Paxson, D. E., "An Improved Numerical Model for Wave Rotor Design and Analysis," AIAA Paper 93-0482, January 1993, (also NASA TM 105915).
3. Paxson, D. E., "A Comparison Between Numerically Modeled and Experimentally Measured Loss Mechanisms in Wave Rotors," *AIAA Journal of Propulsion and Power*, Vol. 11, No. 5, 1995, pp. 908-914, (also NASA TM 106279).
4. Nalim, R. M., and Paxson, D. E., "A Numerical Investigation of Premixed Combustion in Wave Rotors," *ASME Journal of Engineering for Gas Turbines and Power*, Vol. 119, No. 3, 1997, pp. 668-675, also ASME Paper 96-GT-116, June 1996, also, NASA TM 107242.