An Overview of Rayleigh's Diagnostic Outputs

1 Introduction

Rayleigh incorporates a system of diagnostic outputs designed to enable the analysis of simulation results with varying degrees of detail. Broadly speaking, diagnostic outputs in *Rayleigh* are comprised of two components: the diagnostic *type* and the diagnostic *quantity.* The frequency with which each *type* is output, as well as the various *quantities* associated with that diagnostic type are controlled by the user through the *main_input* file (*Rayleigh's* control file, read at run time).

In §2, we discuss general aspects of output control in *Rayleigh.* In §3, we provide an overview of the diagnostic types available within *Rayleigh.* In §4, we provide a table of all available diagnostic quantities, as well as the corresponding *quantity code* used to reference those quantities in the *main_input* file.

2 General Diagnostic Control

The basic behavior of each diagnostic type, namely which quantities are computed, how often, and how many are stored within a file, are controlled in a similar manner. This behavior is controlled by adding lines of the following form to *main_input* file:

```
XXXX_values = 1, 111
XXXX_frequency = 1000
XXXX_nrec = 5
```

The appropriate prefix with which to replace XXXX depends on the diagnostic type and is discussed in §3. The *values* line controls which quantities are output for *this particular diagnostic type.* In this case, radial velocity v_r (quantity code 1) and temperature (quantity code 111) are specified. See §4 for a table of diagnostic quantities supported by *Rayleigh.* The *frequency* keyword determines how many time steps elapse between outputs. In the example above, one output of v_r and temperature would be carried out every 1,000 time steps. Finally, *nrec* specifies how many outputs are stored within a given file. In this case, 5 outputs would be stored within each file. As the code executes, a series of files would appear in the appropriate diagnostic directory. They would be named:

```
00005000
00010000
00015000
etc.
```

Files are numbered according to the *final* time step they contain. The file 00010000 thus contains data from time steps 6,000, 7,000, 8,000, 9,000, and 10,000.
Note: Full 3-D outputs do not support multiple time steps per file.
Note: The ordering of diagnostic quantity codes in the *main_input* file may be anything (e.g., ascending, descending or random). Moreover, that ordering is unrelated to the order in which *Rayleigh* computes diagnostics at runtime.
Note: The *values*, *frequency*, and *nrec* specified for one diagnostic type are unique to that type and do not need to match those specified for other diagnostic types.

3 Diagnostic Types

The diagnostic types available in *Rayleigh*, and their associated subdirectories (in parenthesis), are:

1. Shell Slices (Shell_Slices)
2. Shell Spectra (Shell_Spectra)
3. Point-wise Probes (in development)
4. Azimuthal Averages (AZ_Avgs)
5. Shell Averages (Shell_Avgs)
6. Volume Averages (G_Avgs)
7. 3-D Output (Spherical_3D)

Any combination of these diagnostics may be activated, at the user's discretion, for a given run. Each output type is stored within the indicated subdirectory (located within the main run directory). These subdirectories are created by *Rayleigh* at runtime. All outputs are written in unformatted binary and contain basic header information (e.g., array sizes) in addition to the raw data. A summary of each diagnostic type follows.

3.1 Shell Slices

Shell-slice (i.e., spherical-surface) output also provides a detailed view of the simulation. When shell-slice output is activated, a series of snapshots are taken of the user-specified diagnostic quantities at user-specified radii. This means that, in addition to the standard diagnostics controls, the output radii must also be specified in the *main_input file.* Radii are specified using their associated *radial indicies.* Radial index 1 corresponds to the outer boundary, and radial index N_r corresponds to the lower boundary (where N_r is the number of radial points used for the run).

Shell-slice files contain a basic header, followed by a data cube dimensioned $(1 : N_\phi, 1 : N_\theta, 1 : N_{radii}, 1 : N_q, 1 : N_{rec})$. Here, N_{radii} is the number of radii specified, N_q is number of diagnostic quantities specified, and N_{rec} is the number of time steps stored within each file.

Example Usage: Suppose that you are evolving a simulation with 128 radial points. You would like to output just below the upper boundary, near the middle of the domain, and near the lower boundary. These radial levels might correspond to radial index 2 (1 point in from the upper boundary), radial index 64, and radial index 127 (1 point in from the lower boundary). You would like to output radial velocity v_r (quantity code 1) and temperature (quantity code 111) once every 2,500 timesteps, and you would like to store 2 timesteps per file. The relevant additions to the *output_namelist* in *main_input* would be:

```
shellslice_values = 1, 111
shellslice_levels = 2, 64, 127
shellslice_frequency = 2500
shellslice_nrec = 2
```

3.2 Shell Spectra

Shell-spectra outputs are analagous to the shell-slice outputs, but contain the complex spherical harmonic tranform of the specified diagnostic quantities at the requested radii. These files also contain a basic header, followed by a double-precision, complex data cube dimensioned $(0 : \ell_{max}, 0 : \ell_{max}, 1 : N_{radii}, 1 : N_q, 1 : N_{rec})$. **Example Usage:** Suppose that you are evolving a simulation with 64 radial points. You would like to output just below the upper boundary, and near the middle of the domain. These radial levels might correspond to radial index 2 (1 point in from the upper boundary) and radial index 32. You would like to output radial velocity v_r (quantity code 1) and temperature (quantity code 111) once every 2,000 timesteps, and you would like to store 5 timesteps per file. The relevant additions to the *output_namelist* in *main_input* would be:

```
shellspectra_values = 1, 111
shellspectra_levels = 2, 32
shellspectra_frequency = 2000
shellspectra_nrec = 5
```

3.3 Point-wise Probes (under development)

Revise later

3.4 Azimuthal Averages

The azimuthal average $f(r, \theta)$ of diagnostic quantities $q(r, \theta, \phi)$ may also be output, with

$$f(r,\theta) = \frac{1}{2\pi}\int_0^{2\pi} q(r,\theta,\phi)d\phi. \tag{1}$$

Example Usage: Suppose that you would like to output radial mass flux $\bar{\rho}v_r$ (quantity code 49) and latitudinal mass flux (quantity code 52). You would like to output every 250 timesteps, storing 20 timesteps per file so that file numbers are multiples of 5,000. The relevant additions to the *output_namelist* in *main_input* would be:

```
azavg_values = 49, 52
azavg_frequency = 250
azavg_nrec = 20
```

3.5 Shell Averages

An average over spherical angle $f(r)$ may also be computed for each diagnostic quantity $q(r,\theta,\phi)$ such that

$$f(r) = \frac{1}{4\pi}\int_0^{\pi}\int_0^{2\pi} q(r,\theta,\phi)\sin(\theta)d\phi d\theta. \tag{2}$$

Example Usage: Suppose that you would like to output radial enthalpy flux (code 74), radial kinetic energy flux (code 72), and conductive heat flux (code 77), averaged over spherical shells. You would like to output every 100 timesteps, storing 100 timesteps per file so that file numbers are multiples of 10,000. The relevant additions to the *output_namelist* in *main_input* would be:

```
shellavg_values = 72, 74, 77
shellavg_frequency = 100
shellavg_nrec = 100
```

3.6 Volume Averages

An global average f, taken over the full spherical shell may also be computed for each diagnostic quantity $q(r,\theta,\phi)$ such that

$$f = \frac{1}{V}\int_{r_i}^{r_o}\int_0^{\pi}\int_0^{2\pi} q(r,\theta,\phi)r^2\sin(\theta)d\phi d\theta, \tag{3}$$

where V is the volume of the shell, r_i is the inner radius of the shell, and r_o is the outer radius.

Example Usage: Suppose that you would like to output the volume-averaged kinetic energy (code 81), enstrophy (code 67). You would like to output every 50 timesteps, storing 200 timesteps per file so that file numbers are multiples of 10,000. The relevant additions to the *output_namelist* in *main_input* would be:

```
shellavg_values = 67,84
shellavg_frequency = 50
shellavg_nrec = 200
```

3.7 3-D Output

Full 3-D output is most useful for creating 3-D animated and static visualizations. It can also be useful for generating, after the fact, custom diagnostics not available in *Rayleigh*. 3-D output files are different from other outputs in *Rayleigh* in a few ways. They are the only files in *Rayleigh* that carry no header information, only one quantity is stored per file, and only a single time step may be stored per file. Output is striped as $(1 : N_\phi, 1 : N_\theta, 1 : N_r)$

such that the ϕ-index runs the fastest, and the r-index runs slowest. The diagnostic code associated with the output quantity is appended to the end of the filename is used to identify the contents of the file.

Example Usage: Suppose you are carrying out a visualization run. You would like to generate 3-D snapshots of temperature (quantity code 111), azimuthal velocity v_ϕ (quantity code 33), and enstrophy (quantity code 67) with a cadence of one snapshot for every 100 timesteps. The relevant lines you should add to the *output_namelist* in *main_input* are:

```
full3d_values = 111, 33, 67
full3d_frequency = 100
```

As the code runs, a series of files with appear in the Spherical_3D directory with names such as:

```
00000100_33
00000100_67
00000100_81
00000200_33
00000200_67
00000200_81
etc.
```

The suffix on each filename indicates the diagnostic quantity contained in the file.

4 Diagnostic Quantities

To be implemented. See attached FORTRAN source file for now.