1 Introduction

This lists user inputs typically expected for a seakeeping frequency analysis. They are presented in a logical manner to user. The organization follows the typical subject association from a user perspective. Note that not all of these inputs will be implemented in the current project. But they are included to show what future expansions of the program must accommodate.

2 Numerical Inputs

The user will like to have some keyword to prompt them for a value. The ability to enter comments, which are ignored by the program is a godsend to the user. Don't make inputs dependent on position within the text file. It will drive users mad.

2.1 Complex Numbers

One major element is that the user will need a way to enter a complex number. Complex numbers are usually represented in a rectangular or polar form. The user will probably want to option of using either form.

2.2 Units

There are a variety of units of measurement that can be used. But allowing a program to create multitudes of different units can quickly confuse the program development and the user. I would suggest that we setup the program to only us Standard International (SI Metric) units. Later expansions, such as a GUI, might allow multiple unit systems.

3 Inputs

3.1 Hydrodynamic Database

Some other program did the hard math to calculating the interaction of forces with the body. This program will write out these forces as a list, sorted by wave frequency and wave direction. The format for this list does need to be decided for use by oFreq. Normally, all the forces are output for a single wave height of 1.0 m.

The user will specify the location of the files that compose the hydrodynamic database. This may be a single file, multiple files, or a folder which contains multiple files, depending on how the project develops.

3.2 System Properties

These are global properties that control the behavior of the program. Most of the other settings will control the physics of the math behind the program. These settings instead control the program itself.

3.2.1 Analysis Type

This controls the type of analysis that oFreq will perform.

1. **Response Analysis** – Uses matrix inversion to solve for responses directly.
2. **Resonance Analysis** – Uses Eigen problem matrix to solve for resonant frequencies and mode shapes.

3.2.2 Wave Frequencies

This is a list of wave frequencies. The base SI unit for frequency is radians / second (rad/s).

Normally, this is a list of regularly spaced frequencies, ranging from 0.005 – 3.00 rad/s, or about that magnitude. But sometimes the user will add in a specific frequency. There is nothing in the physics or mathematics that requires a regularly spaced frequency list.

3.2.3 Wave Directions

This considers the various wave directions to analyze. This is measured as an angle, relative to the ship. The definition of the direction depends on the orientation of the coordinate system with the ship. That one can get tricky and we will need to work out exactly what coordinate system we want to use.

Normally, this is a list of regular spaced angles, ranging from 0.00 – 3.14 rad. Sometimes the user will add in a specific wave direction. There is nothing in the physics or mathematics that requires a regularly spaced direction list.

3.3 Body Properties

These are physical constants about the body itself. A body is a general term for the marine structure, be it a ship, an oil platform, or a floating raft. Just anything that sits on the ocean. Body definition gets a little tricky because eventually, the program would expand to allow definition of multiple bodies, each one having its own set of properties.

3.3.1 Name

This would be the name the user would give to the body. Just something for human reference.

3.3.2 Mass

This is the mass for the body. We are assuming each body has 6 degrees of freedom. It has the following properties that are all required for complete mass definition. I included the names and their most common symbol.

1. **Mass (M)**– This is the straight mass of the body, in kilograms (kg).
2. **Mass Moment of Inertia XX (I_{XX})**– This is used to calculate momentum of body rotation about the X-axis, measured in kg-m^2.
3. **Mass Moment of Inertia YY (I_{YY})** – This is used to calculate momentum of body rotation about the X-axis, measured in kg-m^2.

4. **Mass Moment of Inertia ZZ (I_{ZZ})** – This is used to calculate momentum of body rotation about the X-axis, measured in kg-m^2.
5. **Cross Moment of Inertia XY (I_{XY})** – This is used to calculate cross coupling of rotation momentum between X-axis and Y-axis.
6. **Cross Moment of Inertia XZ (I_{XZ})** – This is used to calculate cross coupling of rotation momentum between X-axis and Z-axis.
7. **Cross Moment of Inertia YZ (I_{YZ})** – This is used to calculate cross coupling of rotation momentum between Y-axis and Z-axis.

This gets a little more complicated when we consider generalizing the program. The current inputs I gave you specify mass for a normal 6DOF (6 degree of freedom) situation. But if we want to generalize it any number number of DOF, then the user needs to be able to add that information. There is a later input for custom user defined masses. We can probably use that. But if we generalize it like that, then there could be cases where the user will have no need to define all the mass terms above. In that case, the program will need to avoid throwing errors if no input is found and just provide a default value for that variable (probably 0).

3.3.3 Geometry

This option won't be implemented in the current project. But eventually, a user might want to visualize the body motions. This means that some other program generated a mesh of the body. But the mesh will include other items not part of the body. So the geometry section is a list of integers specifying the panel numbers that should be controlled by the motions for this body.

3.3.4 Initial Position

3.3.4.1 Center Of Gravity

The center of gravity gets used later to calculate certain local body motions. And If there is more than one body, then the center of gravity is necessary to determine the relative position between bodies. Center of gravity requires the following three values.

1. **Centroid X-axis (CG_X)** – Center of gravity, X-axis coordinate, in meters (m).
2. **Centroid Y-axis (CG_Y)** – Center of gravity, Y-axis coordinate, in meters (m).
3. **Centroid Z-axis (CG_Z)** – Center of gravity, Z-axis coordinate, in meters (m).

3.3.5 Equations

I'm not sure if this is a user input or something the program tracks internally. The problem is that each body generates 6 equations of motion. Or possibly more if the user is using some custom solver. So, how do we track what equations go to what body. Because the program will just solve the equations and get a list of motions. We need to then relate that back to the user in terms of bodies.

For example, with 3 bodies and 6 equations each, the program will come up with a solution of motions 1 – 36. We need some way to track that motion number 35 relates

back to the 5th motion of body number 3.

3.4 Wave Environment

The wave environment is what defines the wave height for each frequency and wave direction. Normally, this is defined through a wave spectrum, which specifies the wave energy for each wave frequency. Some math later on converts wave energy to wave height.

3.4.1 Wave Spectrum

3.4.1.1 Custom Spectrum

The custom spectrum will just be a list of wave frequency and wave energy. Not sure if we want to require the user to use the same wave frequencies that they used in Section 3.2.2., or if they can specify a new list.

3.4.1.2 Specified Spectrum

This is the typical option a user will go to. There are several pre-defined spectrums that exist. And each spectrum has some specific parameters that define it. So we need a way for the user to first specify which type of spectrum they want to use (picking from a list of named spectrums), and then specify the custom parameters for each spectrum.

3.4.2 Wave Directions

If you look at ocean waves, you will notice that normal sea has waves coming from all directions. But there is usually a dominant wave direction. This means that the naval architects handle this one of two ways.

3.4.2.1 Option 1: Individual Wave Spectrums

One option will be to let the user specify a series of wave spectrums to use as a library. And then for each wave direction, they would specify which wave spectrum to use.

3.4.2.2 Option 2: Spectrums Derived From A Primary Direction

Most of the time, we use wave spreading functions. That is, we define a single wave spectrum for the primary direction. And then there are certain engineering formulas to specify how that wave spectrum should scale up or down for each wave direction off from the primary direction.

3.5 User Forces

3.5.1 Reaction Forces

These are user defined forces that will depend on the solved equations of motion. An example might be a simple mooring line that holds the the ship at anchor. For each active force, the user needs to be able to define a coefficient, as a real number, and how

that coefficient relates to the equations of motion.

They will need to specify which equation they are using, and the derivative within that equation that the coefficient applies to. For sake of user organization, it will help to be able to group multiple equations all under one force.

3.5.2 Active Forces

These forces are relatively simple to define. The force gets entered as a complex number. Again, the force needs to be specified to an equation of motion. At the most complex, you would specify the equation of motion and then give a list of complex numbers relating to wave frequency.

3.5.3 Derived Forces

This won't be implemented in this project. But I wanted to include it just in case it affects how we structure the input file format. There are certain forces that end up affecting all equations of motion. And their interaction is too complex for the user to calculate ahead of time. These forces, the user would provide the inputs for, and the program would derive the actual inputs as a preprocessor to solving. The main use for such a derived input would be a mooring line. Mooring lines can become very complex. The easiest way for the user is to define the properties of individual mooring line components and then specify the assembly sequence.

3.5.3.1 Mooring Line

Mooring lines have the following components.

1. Point Mass
2. Cable
3. Point Buoyancy
4. Damping Disc

Each gets their own set of properties.

4 Output Control

I don't know we want to give the user the option to turn outputs on and off, or to have the program automatically spit out all the expected outputs, even if that means some files end up blank.

4.1 Global Results

If we just want the program to write out all outputs, no user input is required for global outputs. For information, this is the list of outputs that would normally be generated.

4.1.1 Position
Reports amplitude and phase of motion. A list of complex numbers for every wave frequency. One list for every equation included.

4.1.2 Velocity
Reports velocity and phase of motion. A list of complex numbers for every wave frequency. One list for every equation included.

4.1.3 Acceleration
Reports acceleration and phase of motion. A list of complex numbers for every wave frequency. One list for every equation included.

4.1.4 Higher Derivatives
Reports amplitude and phase of higher derivatives of motion. A list of complex numbers for every wave frequency. One list for every equation included. This list has no physical meaning. But if someone wanted to design a control system for seakeeping, they might want access to higher order derivatives.

4.1.5 Forces
Forces resulting from motions. This may be broken down into individual forces. For example, all the user defined reaction forces could come out as individual items. And the forcing terms on the body.

4.1.6 Energy
Energies resulting from motions. This is useful to people designing wave energy extraction devices.

4.1.7 Efficiency
Efficiency of body motions. This is useful to people designing wave energy extraction devices. This output might require a few user inputs.

4.1.8 Human Metrics
These are some industry based rules to calculate the odds of motion sickness for passengers and other results for human performance.

4.2 Local Results
Sometimes the engineer is concerned about the results at a specific point on the body. For example, if a ship if rocking back and forth violently, you can imagine that will create very severe motions on a bridge 100 feet above the water. Almost all the same results

can be output. They just go through some different math. The only major user input that is required is the coordinates of the local point. (x, y, z coordinates)

1. Position
2. Velocity
3. Acceleration
4. Higher Derivatives
5. Forces
6. Energy
7. Efficiency
8. Human Metrics

4.3 Statistics

Statistical summaries of any of the above outputs. This is probably outside the scope of this project.