Dynamic Model for a simplified spar floater supporting the DTU 10 MW wind turbine

Authors

Varun Channabasappagowda Biradar - s200235
Joshua Penrod Forrest - s212474

November 30, 2021

Part 1: Model Formulation

In this section, the analytical formulation for a floating wind turbine is performed. The main parts are simplified into spar buoy, a tower, a rotor and a nacelle with rigid connections. The mooring is also abridged into a linear spring which provides the necessary restoring force.

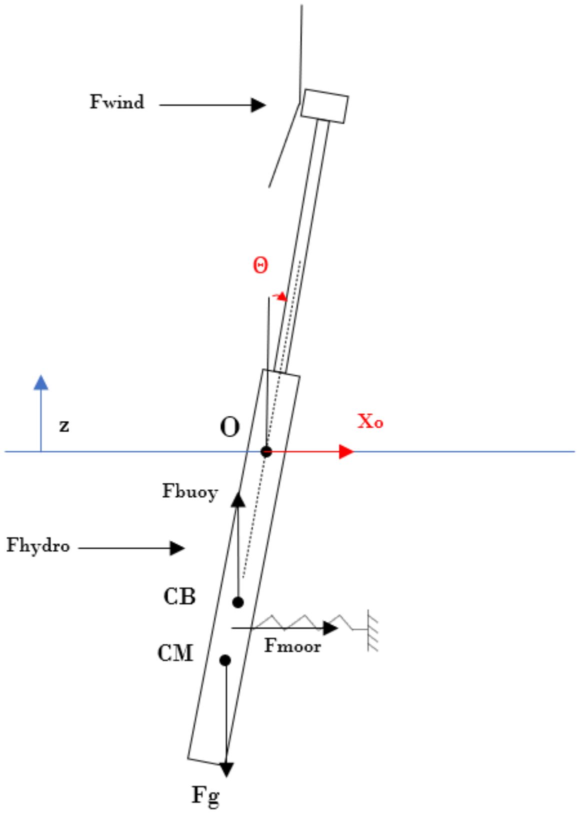

Figure 1: Model formulation diagram.

Question 1: Calculation of Inertial parameters and center of mass

The total mass M_{tot} is calculated as the sum of mass of the turbine structure components. It can be expressed as follows:

$$M_{\text{tot}} = \sum M_{\text{comp}} = M_{\text{floater}} + M_{\text{tower}} + M_{\text{nacelle}} + M_{\text{rotor}} \tag{1}$$

The total mass of the floating wind turbine is $1.2118 \cdot 10^7$ kg.

The center of mass of the system is the superposition of masses and center of masses of its components. This is computed using weighted relative position of the masses of each component given as:

$$
\begin{aligned}
z_{\text{CM,tot}} &= \frac{1}{M_{\text{tot}}} \sum M_{\text{comp}} \cdot z_{\text{CM,comp}} \\
&= \frac{M_{\text{floater}} \cdot z_{\text{CM,floater}} + M_{\text{tower}} \cdot z_{\text{CM,tower}} + M_{\text{nacelle}} \cdot z_{\text{CM,nacelle}} + M_{\text{rotor}} \cdot z_{\text{CM,rotor}}}{M_{\text{tot}}}
\end{aligned} \tag{2}
$$

The center of mass of the system is -86.1109 m, meaning that mass of the structure is concentrated underwater due to heavy spar buoy which helps stabilize the structure.

The moment of inertia around point O (see Figure 1) is calculated using the parallel axis theorem which is the transposition of inertia about the center of mass to another axis parallel to it.

$$
\begin{aligned}
I_O = I_{CM,floater} + M_{floater} \cdot z^2_{CM,floater} + I_{CM,tower} + \\
M_{tower} \cdot z^2_{CM,tower} + I_{CM,nacelle} + M_{nacelle} \cdot z^2_{CM,nacelle} + I_{CM,rotor} \\
+ M_{rotor} \cdot z^2_{CM,rotor}
\end{aligned} \tag{3}
$$

The moment of inertia at the waterline is $1.4566 \cdot 10^{11}$ kg·m^2.

Question 2: Local surge displacement

From Figure 2 representing the translation motion of the turbine-sparbuoy system, the local surge displacement, $x(z)$, can be represented as a function of x_0 and θ. Assuming small angles where $\sin\theta \approx \theta$, the kinematics of each z-position along the turbine/sparbuoy can be transformed into the (x_0, θ) model using the transformation:

$$x(z) = x_0 + z\theta \tag{4}$$

$$\dot{x}(z) = \dot{x}_0 + z\dot{\theta} \tag{5}$$

$$\ddot{x}(z) = \ddot{x}_0 + z\ddot{\theta} \tag{6}$$

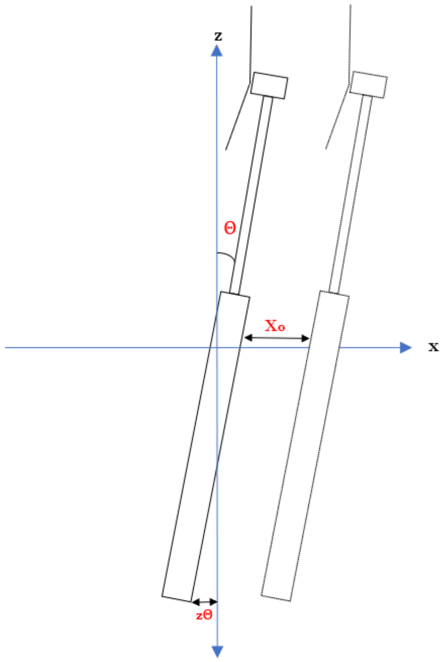

Figure 2: Coordinate transformation diagram.

Question 3: Express F_{moor} and τ_{moor} as a function of (x_0, θ)

The mooring system is considered to be a linear spring system, so the restoring force is proportional to the surge displacement. The surge displacement at the mooring connection height can be found from the transformation given by Equation 4. Since this restoring force acts opposite to the displacement it also is negative.

$$\begin{aligned} F_{moor} &= -K_{moor} \cdot x(z_{moor}) \\ &= -K_{moor} \cdot (z_{moor} \cdot \theta + x_0) \end{aligned} \tag{7}$$

$$\begin{aligned} \tau_{moor} &= F_{moor} \cdot z_{moor} \\ &= -K_{moor} \cdot (z_{moor} \cdot \theta + x_0) \cdot z_{moor} \end{aligned} \tag{8}$$

Question 4: Restoring moment from gravity and hydrostatic forces

The restoring moment from gravity and hydrostatic forces, τ_{Buoy}, is calculated in the model using the case where the system is freely floating with no mooring and no wind or wave forcing. The restoring moment is therefore derived from the balance of the hydrostatic and gravity forces experienced by the system, as well as the system's second moment of area around the waterplane area. In the full restoring matrix for the system, this is represented by term C_5:

$$C_5 = \tau_5 / x_5 = mg\left(z_b = z_g\right) + \rho g I_{11}^A \tag{9}$$

Here, x_5 is the term that represents the pitch, θ. Rearranging Equation 9 and substituting θ for x_5, the restoring moment can be represented the following, where $I_{11}^A = \int x^2 dA$:

$$\tau_{\text{Buoy}} = \theta\left(mg\left(z_b - z_g\right) + \rho g I_{11}^A\right) \tag{10}$$

A positive restoring moment from the buoy assists the system stability. When designing spar-buoy systems, it is standard practice to design the center of gravity below the center of buoyancy. It is possible for the center of gravity to be above the center of buoyancy and for the system to still be stable if the difference is small enough. However, this would likely lead to increased dynamic response from the system.

Question 5: Integrated hydrodynamic forces and moments

The horizontal hydrodynamic force, df, for a section of spar buoy, dz, is given by Equation 11. Here, the first term is the inertia load per section due to added mass and the second term is the Froude-Krylov force both of these together makeup the non-viscous forcing experienced by a submerged body under wave loading. The drag force is the third term. This equation was derived from Morison equation [1].

$$d\tilde{F}_{hydro} = \rho_w\left(C_m A(\dot{u}(z) - \ddot{x}(z))dz + A\dot{u}(z)dz + \frac{1}{2}C_D D(u(z) - \dot{x}(z)) \mid u(z) - \dot{x}(z) \mid dz\right) \tag{11}$$

Here, A is the cross sectional area of the spar-buoy and D is its diameter in a vertical position. Substituting transformations for surge velocity and accelerations from Equations 5 & 6, the sectional hydrodynamic forcing can be written as follows:

$$\begin{aligned} d\tilde{F}_{hydro} = \rho_w(C_m A(\dot{u}(z) - (\ddot{x}_0 + z\ddot{\theta}))dz + \\ A\dot{u}(z)dz + \frac{1}{2}C_D D(u(z) - (\dot{x}_0 + z\dot{\theta})) \mid u(z) - (\dot{x}_0 + z\dot{\theta}) \mid)dz \end{aligned} \tag{12}$$

The equation is simplified by combining two like terms. Also, in the linear model used in this report, the $C_m A\left(\ddot{x}_0 + z\ddot{\theta}\right)$ term is moved to the left hand side and is accounted for in the added mass matrix. However, that is not shown in the analytical representation here.

$$d\tilde{F}_{hydro} = \rho_w((C_m+1)A\dot{u}(z) - C_m A(\ddot{x}_0 + z\ddot{\theta}) \\ + \frac{1}{2}C_D D(u(z) - \dot{x}_0 - z\dot{\theta}) \mid u(z) - (\dot{x}_0 + z\dot{\theta})dz \mid \quad (13)$$

Now, the above equation can be integrated from the limits z_{bot}, the bottom point of spar-buoy to $z = 0$, the mean sea level. Thus, the total hydrodynamic forcing on the submerged structure is given as:

$$\tilde{F}_{hydro} = \int_{z_{bot}}^{0} d\tilde{F}_{hydro} dz \quad (14)$$

By multiplying Equation 11 by the sectional coordinate z we can estimate the hydrodynamic moment to be:

$$\tilde{\tau}_{hydro} = \int_{z_{bot}}^{0} d\tilde{F}_{hydro} \cdot z dz \quad (15)$$

Question 6: Simplified, linear model for surge and pitch

The surge and pitch of the system can be modeled in a simplified linear system (Equation 16) where the mooring forces are linearized and considered in the restoring matrix, C. Additionally, the linearized force and moment equations for wind and hydrodynamic forcing are used as well.

$$\left(\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} + \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}\right) \begin{bmatrix} \ddot{x}_0 \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} x_0 \\ \theta \end{bmatrix} = \begin{bmatrix} F_{\text{Hydrodynamic}} + F_{\text{Wind}} \\ \tau_{\text{Hydrodynamic}} + \tau_{\text{Wind}} \end{bmatrix} \quad (16)$$

Each of the terms in Equation 16 can be calculated for motion mode: surge and pitch. These are summarized below.

Mass Matrix:

$$M_{11} = M_{\text{Tot}} = 1.2118 \cdot 10^7 \text{ kg}$$

$$M_{12} = M_{21} = M_{\text{Tot}} \cdot z_g = -1.0435 \cdot 10^9 \text{ kg·m}$$

$$M_{22} = I_{33}^b + I_{11}^b = I_O = 1.4566 \cdot 10^{11} \text{ kg·m}^2$$

Added Mass Matrix:

$$A_{11} = \int_{z_{\text{bot}}}^{0} \rho \frac{\pi}{4} D^2 C_m dz = -\frac{\pi}{4}\rho D^2 C_m z_{\text{bot}} = 1.2118 \cdot 10^7 \text{ kg}$$

$$A_{22} = \int_{z_{\text{bot}}}^{0} \rho \frac{\pi}{4} D^2 C_m z^2 dz = -\frac{\pi}{12} \rho D^2 C_m z_{\text{bot}}^3 = 5.8166 \cdot 10^{10} \text{ kg}\cdot\text{m}^2$$

$$A_{12} = A_{21} = \int_{z_{\text{bot}}}^{0} \rho \frac{\pi}{4} D^2 C_m z dz = -\frac{\pi}{8} \rho D^2 C_m z_{\text{bot}}^2 = -7.2708 \cdot 10^{8} \text{ kg}\cdot\text{m}$$

Restoring Matrix:

$$C_{11} = -F_{\text{moor}}/x_O = K_{\text{moor}} = 6.6700 \cdot 10^4 \text{ N/m}$$

$$C_{12} = -\tau_{\text{moor}}/x_O = K_{\text{moor}} \cdot z_{\text{moor}} = -4.0020 \cdot 10^6 \text{ N}$$

$$C_{21} = -F_{\text{moor}}/\theta = K_{\text{moor}} \cdot z_{\text{moor}} = -4.0020 \cdot 10^6 \text{ N/rad}$$

$$C_{22} = C_5 - \tau_{\text{moor}}/\theta = \left(\rho g I_{11}^A + mg\left(z_b - z_g\right)\right) + K_{\text{moor}} \cdot z_{\text{moor}}^2 = 3.3519 \cdot 10^9 \text{ N}\cdot\text{m/rad}$$

Question 7: Natural Frequencies for surge and pitch

Since we are computing the natural frequency of the structure the damping matrix is zero and the external forcing matrix is also zero. Thus the equation of motion can be represented as:

$$\mathbf{(M{+}A)}\ddot{x} + \mathbf{C}x = 0 \tag{17}$$

Now considering a sinusoidal nature for displacement:

$$x(t) = \phi e^{j\omega t}$$

Substituting, the equation of motion becomes:

$$\left(-\omega^2(\mathbf{M}{+}\mathbf{A}) + \mathbf{C}\right)\phi e^{j\omega t} = 0$$

Therefore, the equation can be further simplified into a quadratic expression whose solution is given by eigenvalue analysis.

$$\left(-\omega^2(\mathbf{M}{+}\mathbf{A}) + \mathbf{C}\right) = 0 \tag{18}$$

The two non trivial solutions are $\omega_1^2 \,\&\, \omega_2^2$ which are differentiated by the $\phi_1 \,\&\, \phi_2$ eigenvectors. Hence, the natural frequency for surge and pitch can be found by:

$$f_n = \frac{\sqrt{\omega_n^2}}{2\pi}$$

Thus, the natural frequency of surge is estimated to be $f_1 = 0.0083$ Hz and for the pitch $f_2 = 0.0326$ Hz.

Part 2: Dynamic Analysis

In this section, the linear model is implemented for dynamic response to wind and wave forcing. The total time simulation studied for all responses 1600s and the first 1000 seconds are considered to be transient and next 600s is stable.

Question 8 : First order system with a state vector

The linear model for dynamic response given in Equation 16 can be re-written as a first-order system by using the state vector:

$$\underset{\sim}{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} x_0 \\ \theta \\ \dot{x}_0 \\ \dot{\theta} \end{bmatrix} \tag{19}$$

First, the state variables can be substituted into Equation 16:

$$\left(\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} + \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \right) \begin{bmatrix} \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} F_{\text{Hydrodynamic}} + F_{\text{Wind}} \\ \tau_{\text{Hydrodynamic}} + \tau_{\text{Wind}} \end{bmatrix} \tag{20}$$

Then, rearranging and solving for $\dot{q}_3$ and $\dot{q}_4$:

$$\begin{bmatrix} \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \left(\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} + \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \right)^{-1} \left(\begin{bmatrix} F_{\text{Hydrodynamic}} + F_{\text{Wind}} \\ \tau_{\text{Hydrodynamic}} + \tau_{\text{Wind}} \end{bmatrix} - \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \right) \tag{21}$$

This means that for every known pitch and surge, the second time derivative of pitch and surge can be determined by the model.

Question 9: Introducing linear damping

A linear damping term, $B_{11}\dot{x}_0$ can be added to the model for surge where $B_{11} = 2 \cdot 10^5$ N/(m/s). Temporarily neglecting the hydrodynamic and wind forcing, this adjusts the model to become:

$$\begin{bmatrix} \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} = \left(\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} + \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \right)^{-1} \left(- \begin{bmatrix} B_{11} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} q_3 \\ q_4 \end{bmatrix} - \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \right) \tag{22}$$

Question 10: Decay tests for surge and pitch with no hydrodynamic or wind forcing

(a) Initial surge of 1 m: $\underset{\sim}{q} = [1\,0\,0\,0]^T$

The first decay test introduces a unit surge displacement as an initial condition. The system is then released and the time-series response as well as the frequency response is analyzed in Figures 3 and 4. Only one response frequency is seen for surge. The same frequency is the primary response for pitch with a secondary response at a higher frequency. The two response frequencies are very similar to the estimated frequencies from the eigenvalue analysis. The decreasing amplitude of the time response of the structure shows the effect of the damping term on the model.

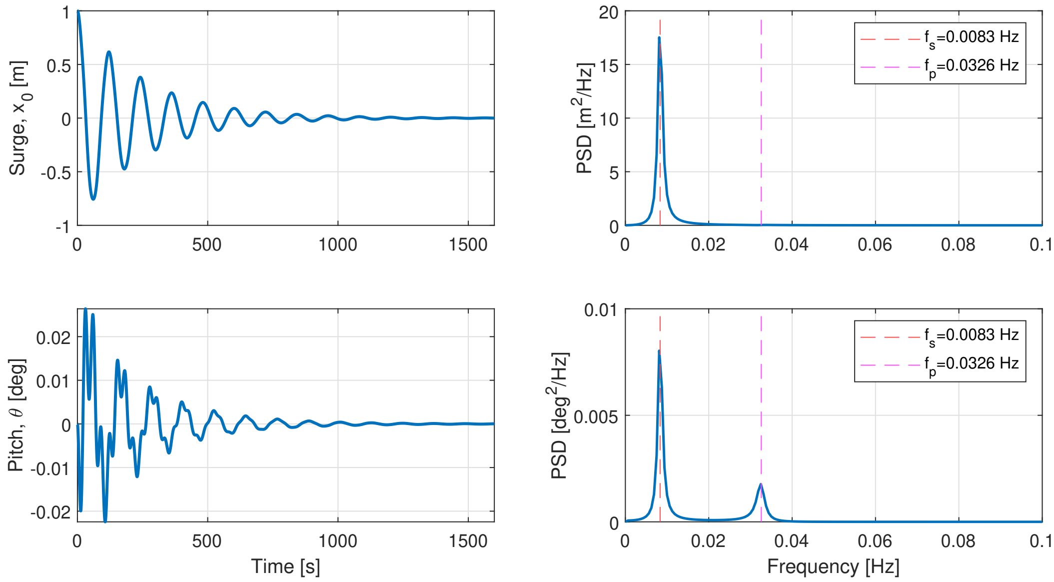

Figure 3: Plots showing the time-series and frequency response to an initial surge of 1 m. The left-hand plots show the time series decay responses for the turbine-floater-spar-buoy system surge (top) and pitch (bottom). The right-hand side plots show the frequency of response.

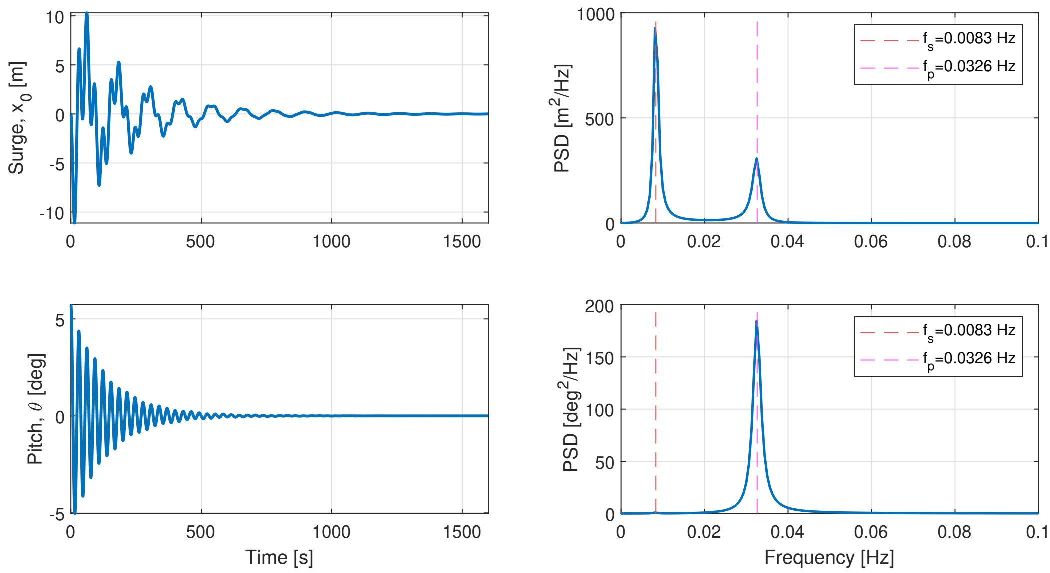

Figure 4: Plots showing the time-series and frequency response to an initial pitch of 0.1 radians. The left-hand plots show the time series decay responses for the turbine-floater-spar-buoy system surge (top) and pitch (bottom). The right-hand side plots show the frequency of response.

(b) Initial pitch of 0.1 radians: $\underline{q} = [0\,0.1\,0\,0]^T$

When the initial condition is a 0.1-radian-pitch displacement, the system's dynamic response mirrors the surge response. However, in this case, the surge response has a single response frequency and the pitch response has two frequencies: a primary frequency response at the same frequency as the surge displacement case, and a secondary frequency response that matches the frequency of the pitch. Comparing Figures 3 and 4, the lower of the two observed frequencies is the surge frequency and the higher of the two frequencies is the pitch frequency. Given that each motion mode experiences both its frequency and the frequency of the other motion mode when the other mode is perturbed, it can be concluded the pitch and surge motions are coupled.

Question 11: Decay tests for surge and pitch with hydrodynamic forcing

By including the hydrodynamic forcing expressed in Equation 11 we can re-simulate the decay tests done in question 10 and compare the results. These hydrodynamic forcing terms add onto the damping matrix as they are essentially hindering the motion of the structure under water by dissipating energy.

(a) Comparing initial surge of 1 m

By giving an initial surge displacement of 1 m and releasing the motion of the floater is analysed. Simulating with this initial condition the equation of motion is solved by *ode4.m*.

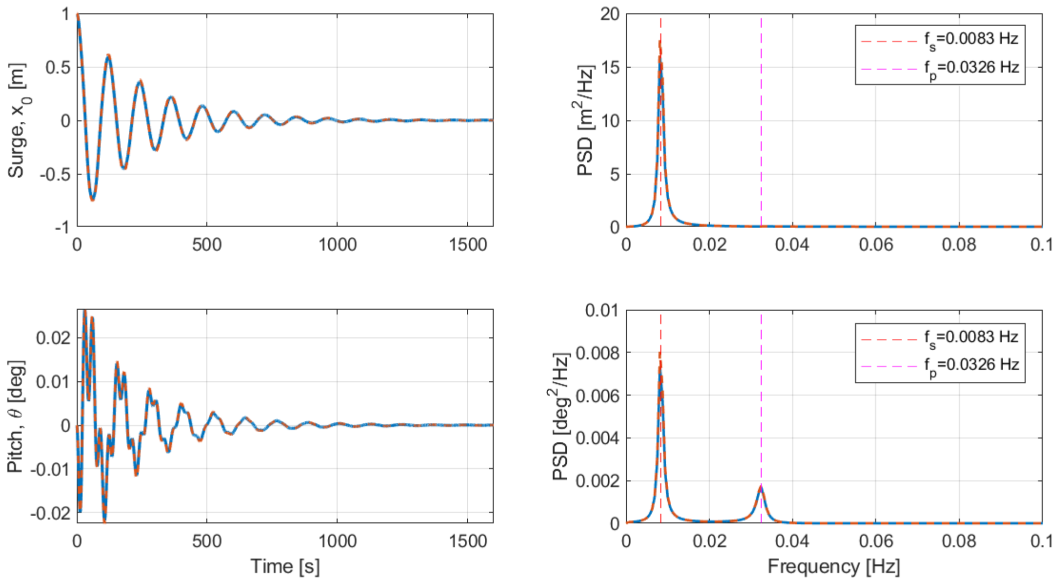

Figure 5: Surge condition with(blue) and without hydrodynamic forcing(dotted orange).

The response obtained with hydrodynamic damping is plotted over the result from question 10. It can be observed that both conditions respond similarly for an initial surge displacement. We see the same coupled response in pitch when surge is excited. Meaning that the coupling effect comes from the diagonal term of the damping matrix as expected. The similarity in the two cases suggests that the introduction of hydrodynamic forcing as a damping effect is not that significant when compared to the restoring forces by mooring systems. There is only a small change in the PSD peak for surge $<$10 % from 17.525 m^2/Hz to 16.06 m^2/ Hz.

(b) Comparing initial pitch of 0.1 radians

By giving an initial pitch displacement of 0.1 rad and releasing the motion of the floater is analysed. Simulating with this initial condition the equation of motion is solved by *ode4.m*.

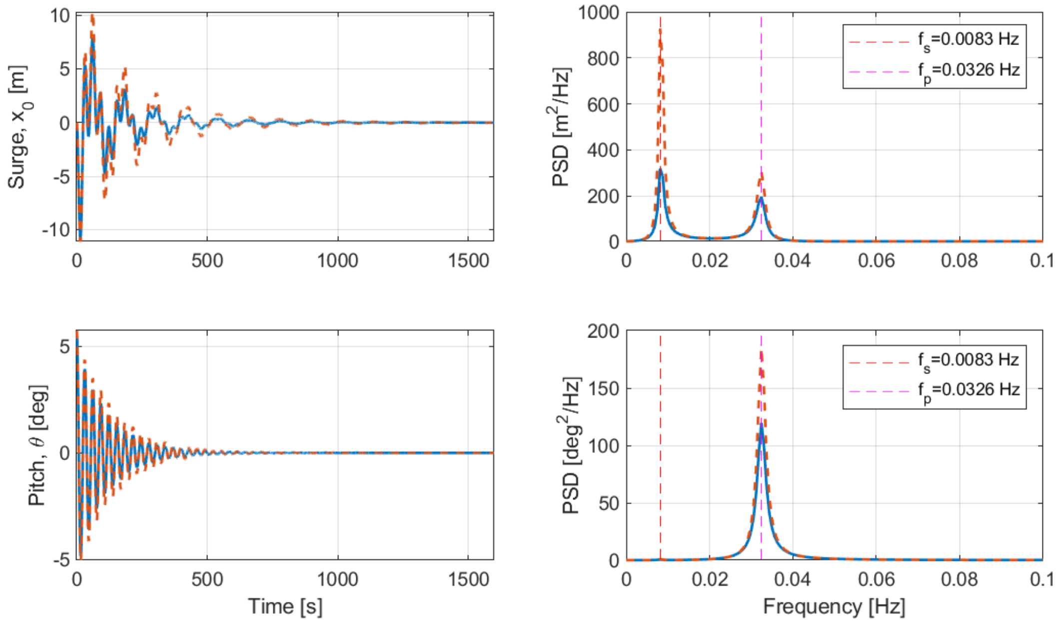

Figure 6: Pitch condition with(blue) and without hydrodynamic forcing(dotted orange).

Now by instigating a pitch of approximately 5 degrees and releasing the motion of the structure is simulated and plotted over the results without forcing from earlier question. The responses are similar again but the magnitudes are reduced for the case including hydrodynamic forcing. An all-around 30 % reduction is noticed. Thus, the inference is that the hydrodynamic forcing is able to provide additional damping effect when the structure submerged is in a pitch-dominant motion. This is not the case for surge initial condition, and likely part of the small difference seen is due to the motion coupling.

(c) Instigating an initial pitch of 1 radian

A pitch decay test is conducted now for an initial condition which gives a large pitch of 1 radian. Here, the hydrodynamic forcing is disabled once again to purely study the damping effect of mooring system and compared with it enabled.

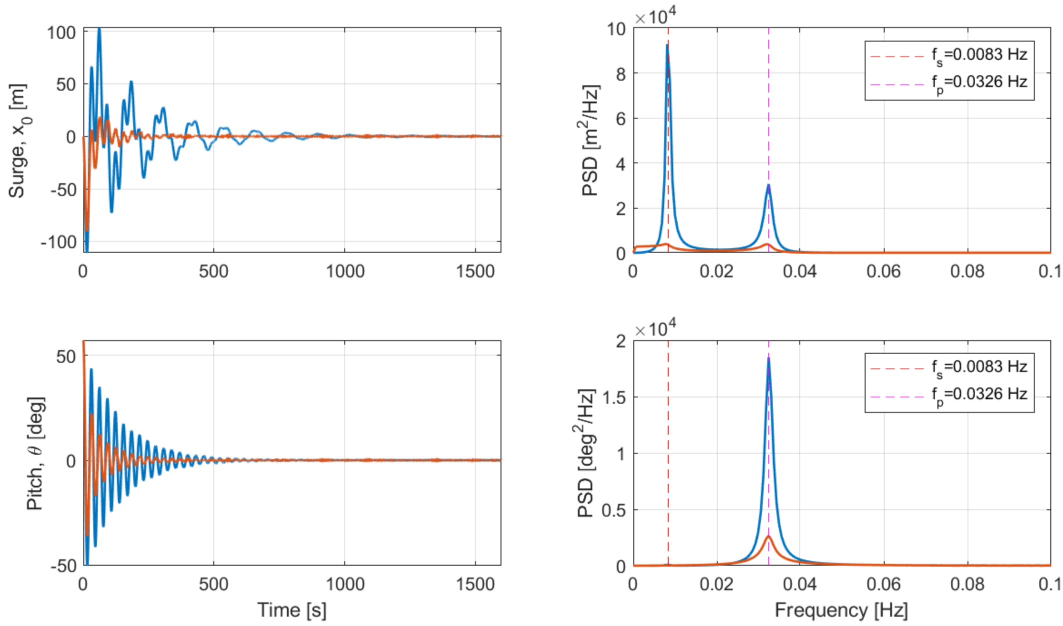

Figure 7: Pitch condition with(orange) and without hydrodynamic forcing(blue).

It is evident now that since the pitch condition is 1 radian a purely hydro static analysis of the response yields in highly unrealistic outputs of surge in excess of 50m. Since our model assumed small angles for θ the surge displacement given by Eq.4 no longer holds valid for a 50+ degree initial pitch. This would require a more extensive modelling for surge displacement one without such a small angle assumption. Including the hydro forcing it is observed that the damping effect due to it is still realistic in terms of pitch response. But the surge response is still off as it reports a surge of almost 80m.

Question 12: Forcing tests for surge and pitch with only regular wave forcing

In this case, the dynamic response from regular wave forcing is considered. The regular waves have the properties $H = 6$ m and $T = 10$ s. For each of the calculations, the wave kinematics are computed assuming an upright buoy position. In Figure 8, the time-response shows that in the steady-state condition, there is a constant oscillation of both surge and pitch and the damping is only able to reduce the amplitude of the oscillations to 1.5 m for surge and approximately 0.65 degrees for pitch. These are very small oscillations, however. Additionally, the primary frequency response is 0.1 Hz, which is the forced response frequency from the waves. There is no observable response in either the surge or pitch's natural frequencies.

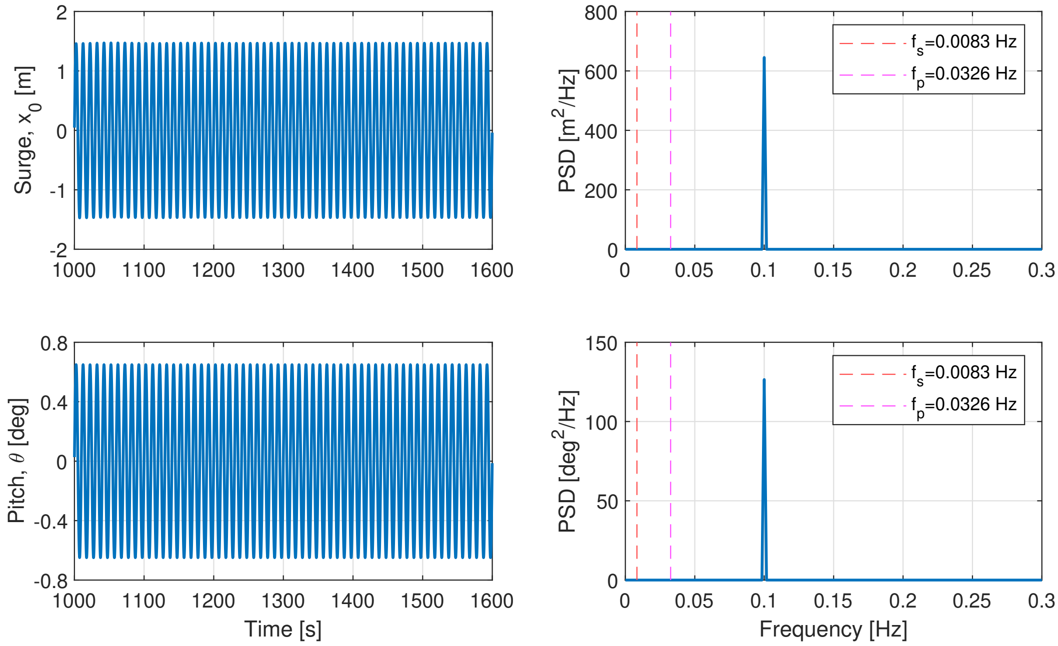

Figure 8: With regular wave forcing, the structure experiences regular oscillations of both surge and pitch. The frequency of both responses is 0.1 Hz, identical to the forcing frequency.

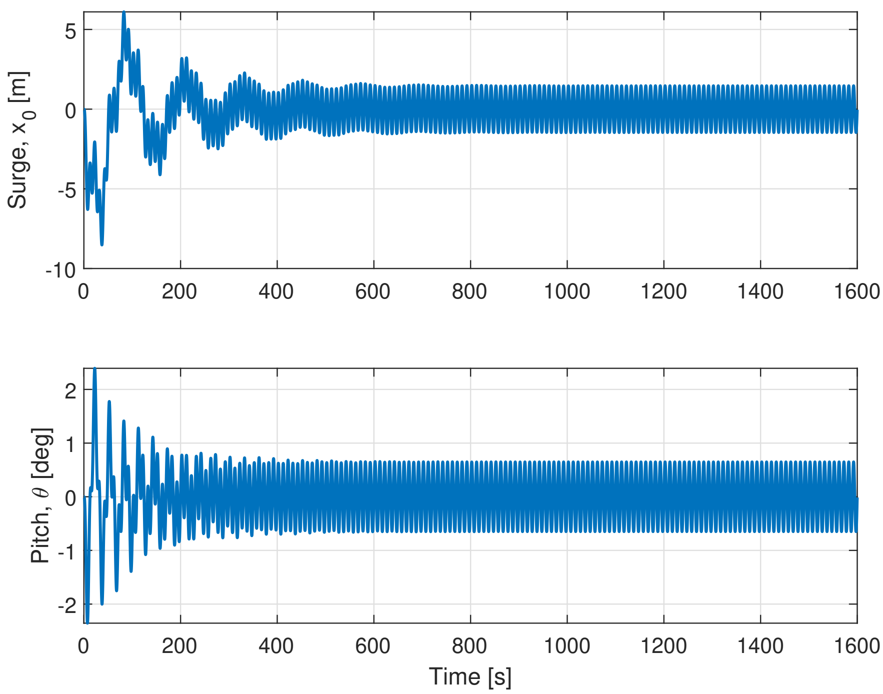

Figure 9: Full time series response for regular wave forcing.

Statistic	Surge [m]	Pitch [deg]
Mean	0.1042e-4	-0.0275e-4
Standard Deviation	1.0372	0.4592
Maximum	1.4714	0.6494
Minimum	-1.4723	-0.6495

Table 1: Summary of statistics for Surge and Pitch with only regular wave forcing.

The statistics presented above it can be seen that pitch and surge have zero mean value with almost the same max and min values indicating steady response. This can be interpreted as the loading due to regular waves is minimal and hence the oscillations reflect the same.

Question 13: Forcing tests for surge and pitch with regular wave and steady wind forcing

In addition to regular waves a steady wind of $8m/s$ is introduced as wind forcing. The response plots and PSD plots are shown below. with the exclusion of first 1000s as transient time from both the frequency domain and the time series. The statistics for steady state time domain also also tabulated below.

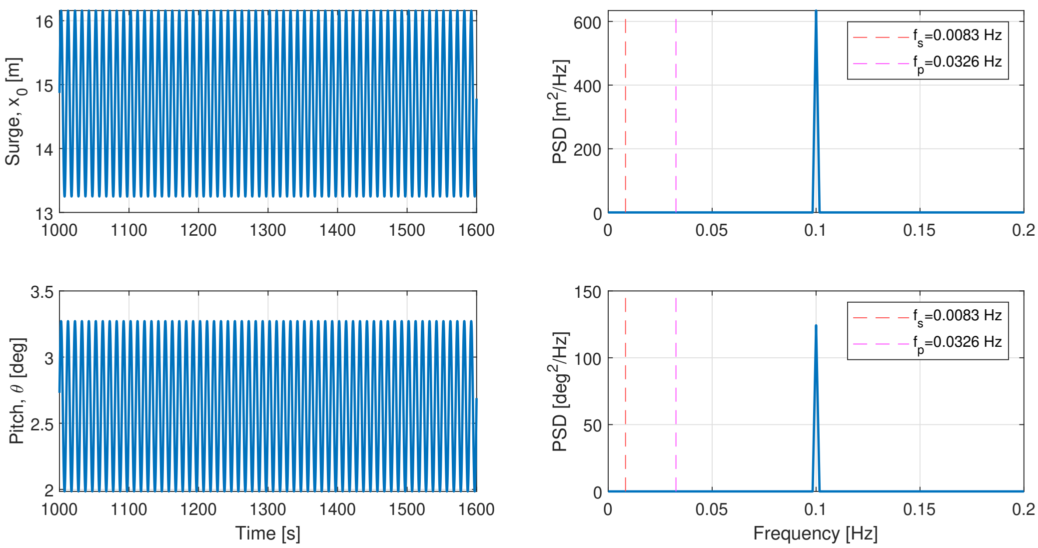

Figure 10: Time series and PSD for regular waves with steady wind of 8 m/s.

With the mean steady wind of 8 m/s comes the rotor thrust forcing. This forcing calculation is done using the global thrust coefficient C_T. First a mean thrust force is calculated and then a time varying thrust time series is computed and the resultant time series for forcing is determined via an adhoc reduction of dynamic part to compensate for turbulence as directed in the report hints [2].

Here, it is observed that the although the standard deviation is same as that of question 12 there is noticeable increase to mean, maxima and minima for surge and pitch both. This is attributed to the fact that the mean loads for wind are comparatively more significant and hence influence the response of the structure more than the wave loading alone and also that the wind forcing unlike wave is constant and not sinusoidal in nature blowing in the same direction. As for the PSD plot the excitation frequency is at 0.1 Hz as this is still the forcing frequency due to wave conditions.

Statistic	**Surge [m]**	**Pitch [deg]**
Mean	14.7020	2.6272
Standard Deviation	1.0282	0.4552
Maximum	16.1587	3.2718
Minimum	13.2487	1.9846

Table 2: Summary of statistics for Surge and Pitch with regular wave and steady wind forcing.

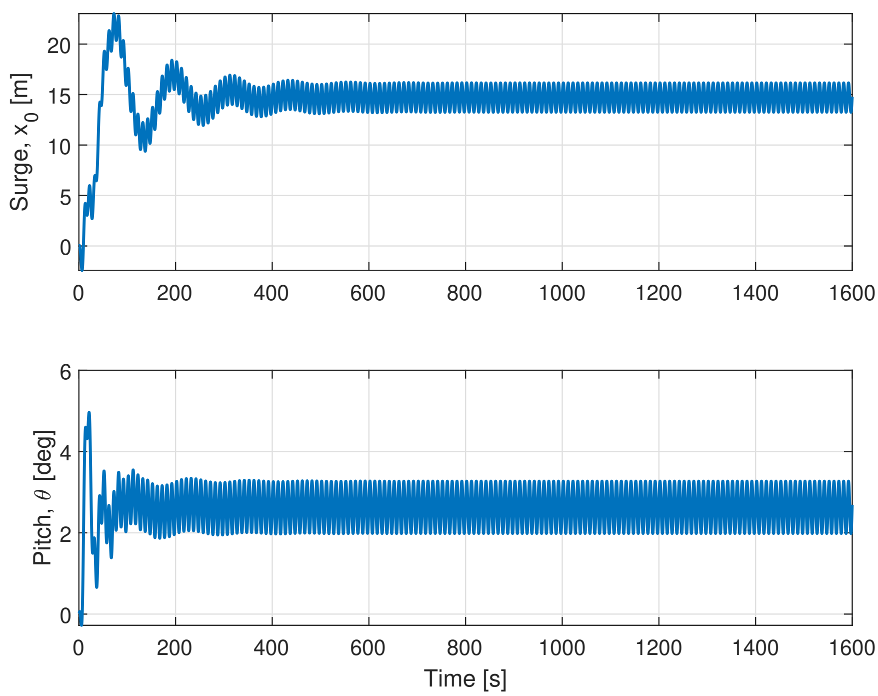

Figure 11: Full time series response for regular wave forcing with steady wind of 8 m/s.

When comparing the above figure with Figure 9 it can be observed that the pitch transient signal is much more different in that the steady state is reached much faster when a constant wind forcing is introduced this is due to the aerodynamic damping that helps restore stable response more quickly.

Question 14: Forcing tests for surge and pitch with irregular wave and steady wind forcing

This analysis considers the structure's dynamic response to forcing from irregular waves and steady wind. A JONSWAP spectrum is used to simulate a time-series case of irregular wave kinematics. The parameters used for the JONSWAP spectrum are: $H_S = 6$ m, $T_P = 10$ s, and $\gamma = 3.3$. The same 8 m/s steady wind speed is used.

The steady-state time-series and frequency response is presented in Figure 12. The average surge is approximately 14.5 m and the average pitch is approximately 2.5 degrees. From Figure 8, it can be seen that the response to waves is oscillatory and centered around 0. Additionally, Figure 10 shows the effect of steady wind forcing on surge and pitch. Therefore, the values of the average surge and pitch are due to wind forcing, and the oscillations are from the irregular wave forcing. The frequency response shows a peak frequency of 0.1, which is consistent with the regular wave forcing due to the same H and T parameters being used. The relative magnitude of the 0.1 Hz frequency peak is higher for the pitch response than the surge response. This shows the effect of the wind damping on the surge of the system.

However, frequencies are also consistently observed from 0.05 Hz to 0.15 Hz for both pitch and surge. This is due to the irregular waves within the sea state having similar but not regular frequencies. There is a small frequency response at the surge frequency, but this is likely due to surge-pitch coupling.

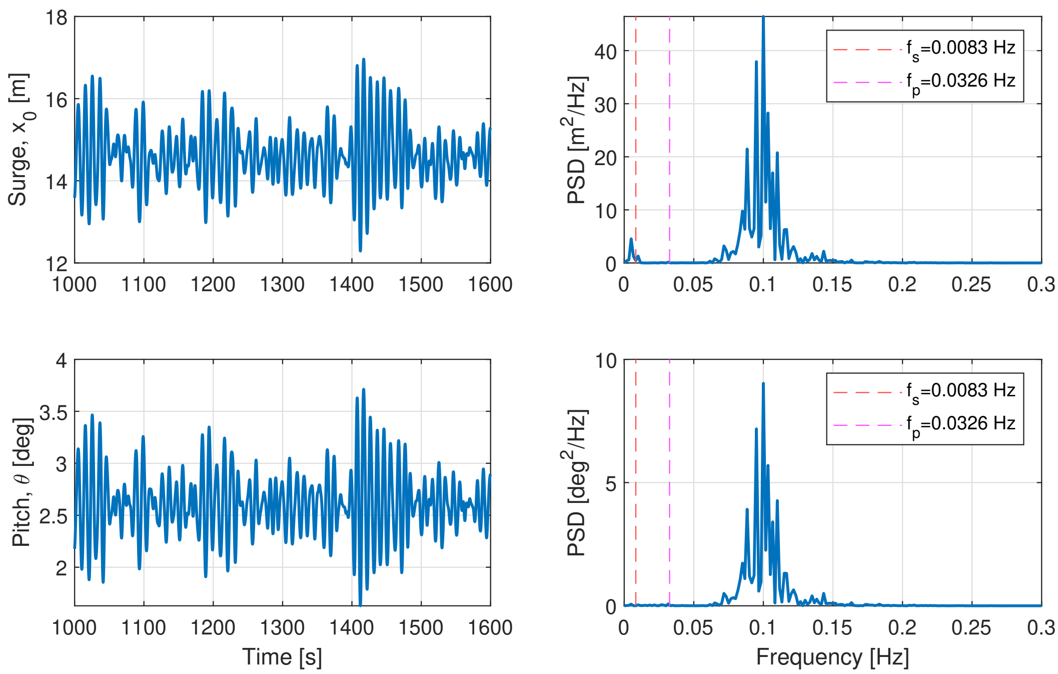

Figure 12: Steady-state time-series and frequency response of surge and pitch with irregular wave and steady wind forcing.

Statistic	Surge [m]	Pitch [deg]
Mean	14.5715	2.6044
Standard Deviation	0.6006	0.2595
Maximum	16.4920	3.4431
Minimum	12.9976	1.8799

Table 3: Summary of statistics for Surge and Pitch with irregular wave and steady wind forcing.

Question 15: Forcing tests for surge and pitch with irregular wave and unsteady wind forcing

In this question wind forcing is altered from being steady to unsteady with the introduction of Kaimal wind time-series based on $V_{10min} = 8$ m/s , $I = 0.14$ and $l = 340.2$ m. The time and frequency response are shown in below figure excluding the transient signal.

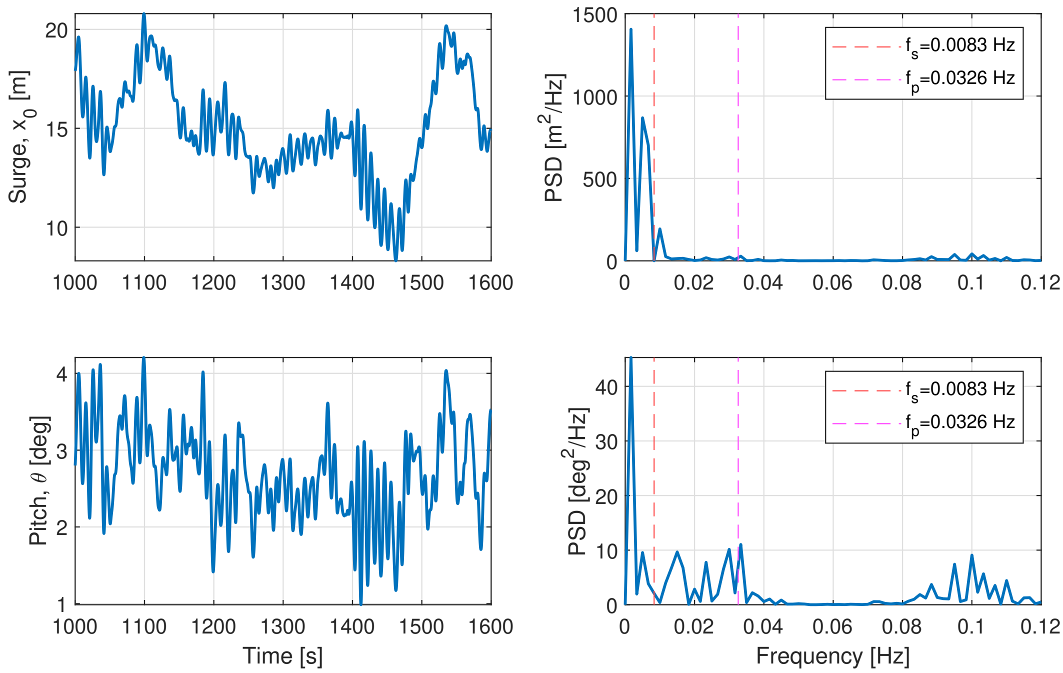

Figure 13: Dynamic response to irregular wave and unsteady wind forcing. The surge response is primarily driven by wind forcing where the pitch response is driven by wind forcing, surge coupling, and some wave forcing.

It can be observed from the above figure in conjunction with the statistics for forcing with irregular wave and unsteady wind that the oscillations for both times series have further increased owing to the turbulent nature of the wind forcing thus reflecting a significant change in standard deviation, maxima and minima when compared to the 3 previous forcing tests from question 12 through 14. The PSD plots also reflect a similar behaviour in that the surge peaks now observed at the very low end of the frequencies due to the Kaimal spectrum of wind(0.0016 Hz) which also falls in this zone and natural frequency of the surge is also instigated. For the pitch PSD it is noticeable that it is piqued by both wind and wave forcing once at the Kaimal spectrum region(0.0016 Hz) and again at the natural frequency of pitch as well. There is also some small peaks at the 0.1 Hz due to wave forcing frequency ($1/T_P$).

Statistic	Surge [m]	Pitch [deg]
Mean	14.9844	2.6791
Standard Deviation	2.5013	0.5767
Maximum	20.7973	4.2064
Minimum	8.3008	0.9868

Table 4: Summary of statistics for Surge and Pitch with irregular wave and unsteady wind forcing.

Part 3: Blade Pitch Control for Dynamic Stability

Question 16: Instantaneous controller for steady wind and no wave forcing

In this analysis, there are two load cases. The first is a steady wind of 10 m/s and the second is a steady wind of 16 m/s. The first load case is less than the turbine's rated wind speed while the second is greater than the rated wind speed. There is no wave forcing. The steady-state response for each case is shown in Figures 14 and 15.

Figure 14 shows the response for the steady $V_{10\ \mathrm{min}} = 10$ m/s case. While there is a response oscillation for surge and pitch, the amplitudes are very small because for below rated wind speeds, the thrust increases with the wind speed. Therefore, when the wind speed increases, the thrust increases and the turbine hub moves backwards, lowering the relative velocity and thrust. The turbine then moves forward due to the decrease in thrust. This is a stable dynamic scenario and is depicted in the figure. The frequency response shows that the response is primarily in surge, since the lower frequency has a much higher occurrence.

By contrast, Figure 15 shows a dynamically unstable response for the steady $V_{10\ \mathrm{min}} = 16$ m/s case. The unstable response is in the pitch mode, because that is the dominating frequency response. The turbine is oscillating between -0.5 degrees and just under 6 degrees of pitch and is surging in response to the pitch changes. This rocking motion is due to the turbine's control scheme, which decreases the thrust with increasing wind speed when the wind speed is greater than the rated wind speed. Like in Figure 15, the turbine ends up rocking back and forth, because its motion and changing thrust are opposite signs of each other.

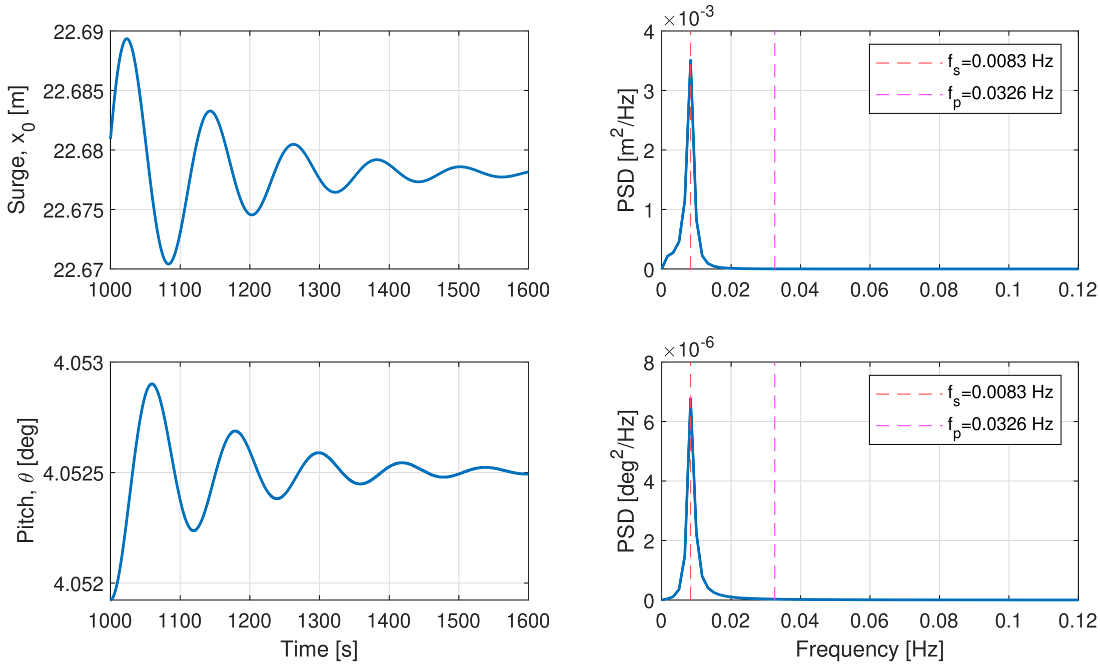

Figure 14: Steady-state response to steady wind forcing using an instantaneous controller for $V_{10\text{ min}} = 10$ m/s, less than rated wind speed.

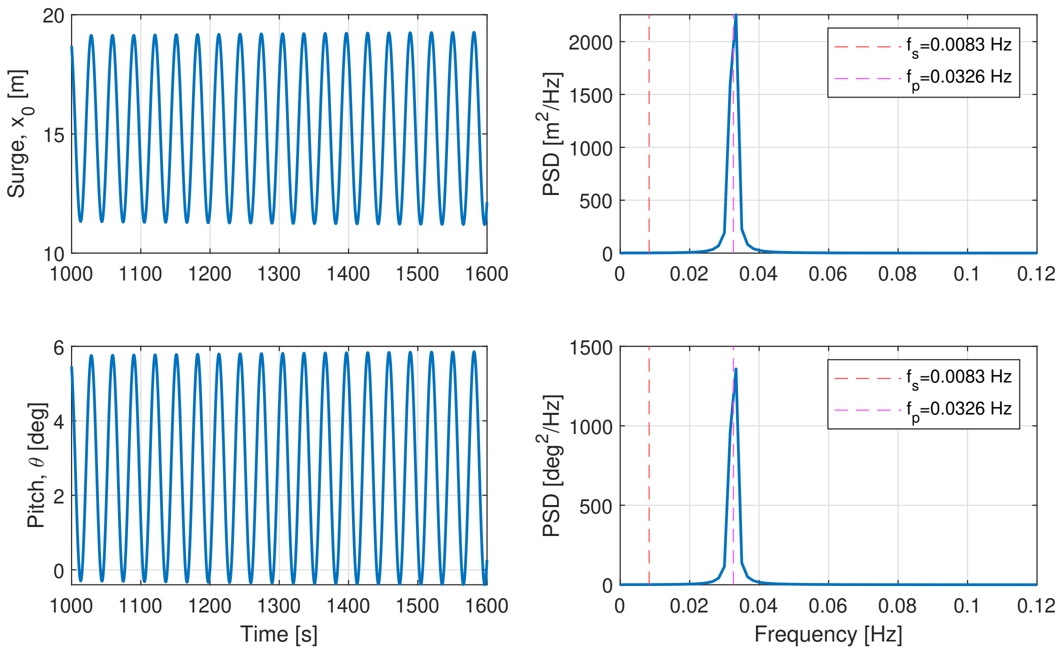

Figure 15: Steady-state response to steady wind forcing using an instantaneous controller for $V_{10\text{ min}} = 16$ m/s, greater than rated wind speed.

Question 17: Adapted floating controller for steady wind and no wave forcing

The previous case with a steady wind speed greater than the rated wind speed demonstrates the need for a time-delayed controller to prevent a dynamically unstable motion response. This can be achieved by changing the control scheme for the turbine above rated wind speeds. A simplified version of a common control model is presented in Equation 23.

$$\frac{dC_T}{dt} = -\gamma \left(C_T - C_T\left(V_{\text{rel}}\right)\right) \tag{23}$$

In the equation, C_T is the "current" C_T where $C_T\left(V_{\text{rel}}\right)$ is what the C_T would be for an instantaneous C_T control scheme for the "current" relative velocity. The difference between these two terms is scaled by a constant factor, γ, to inform the update term for the next C_T, $\frac{dC_T}{dt}$. In practice, the value for γ is dependent upon the magnitude of the time step, Δt. γ generally controls the speed of the controller feedback, and can be adjusted to dampen the oscillatory motion from the instability deriving from instantaneous changes in the C_T. However, as can be seen from the results of the control scheme in Figure 16, $\gamma = 2$ is the wrong choice for a time step of 0.05 seconds. The response is still too fast, and the response of the turbine is very similar to the case in Figure 15 with the same steady wind speed and instantaneous control response.

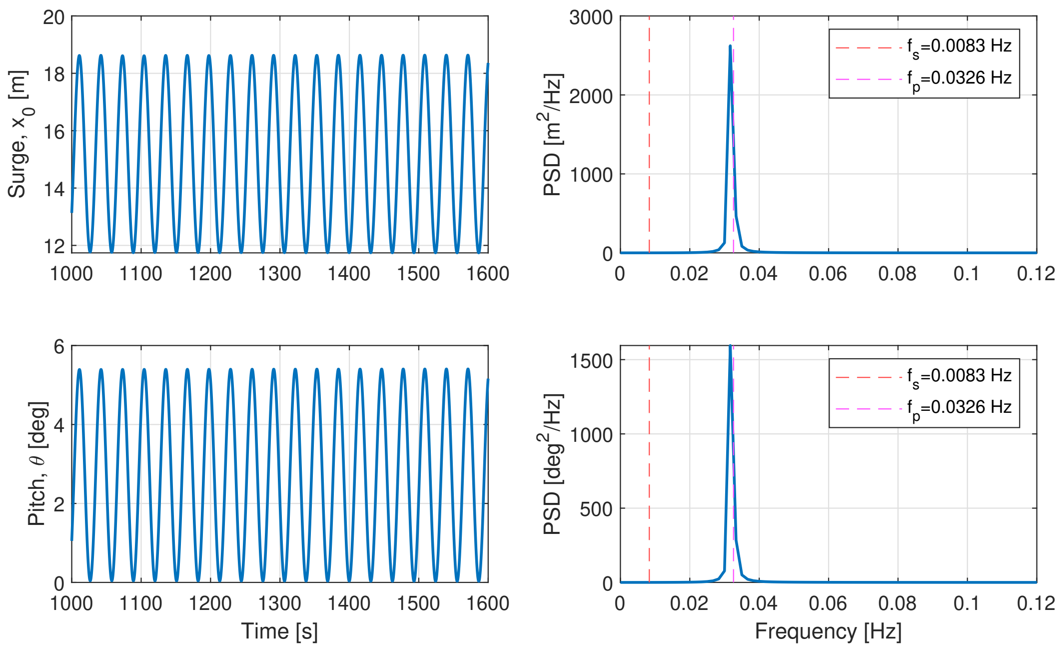

Figure 16: Dynamic response to $V_{10\ \text{min}} = 16$ m/s with adapted thrust controller from Equation 23.

Question 18: Optimizing the adapted floating controller

Three values of γ are presented as a case study to compare control strategies. In order, these values are **0.1**, **0.5**, and **1.5**. The lower the gamma value, the lower the rate of change of C_T in time. As expected, the lowest value of γ produces the most "stable" response of the three (Figure 17). It appears that all three approach stability, but the case where $\gamma = 1.5$ still shows a 2 meter surge amplitude and 2 degree pitch amplitude, so it is probably not time-delayed enough. However, the case where $\gamma = 0.1$ is not necessarily the best case either. The original reason for the control scheme is to reduce the blade loading to extend the turbine's lifetime. Having a value of γ that is too low could reduce a turbine's ability to effective reduce the loads experienced by the blades in time. Therefore, a value of γ which balances these two needs, possibly $\gamma = 0.5$, is the best case scenario. It allows the turbine to reduce the loads in high wind speeds effectively enough while also providing a slow enough response to dampen response oscillations.

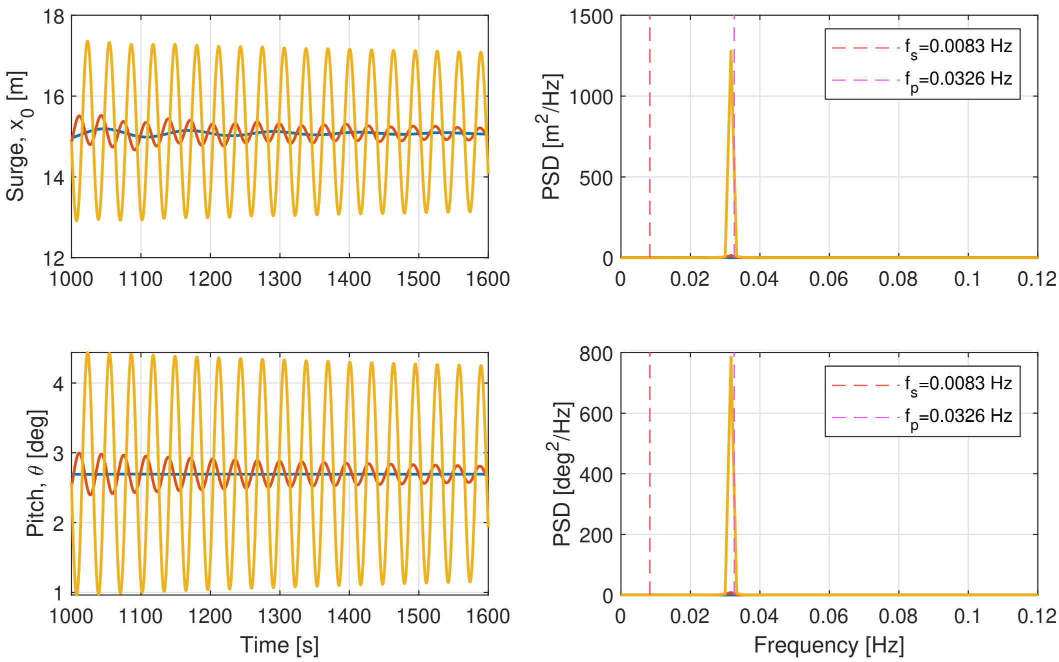

Figure 17: Steady-state response for the adapted pitch control where the wind is steady $V_{10\ \mathrm{min}} = 16$ m/s. $\gamma = 0.1, 0.5, 1.5$ respectively for the blue, orange, and yellow traces.

References

[1] "Froude–krylov force - wikipedia."

[2] *46211 Offshore Wind Energy - Fall 2021 Assignment 5: Dynamic model for a simplified spar floater supporting the DTU 10MW wind turbine.* Technical University of Denmark.

MATLAB Code

Main Code

```
% Assignment 5Josh & Varun
clc; clear all
% Loading the given constants for the floater
global rhow Cm CD D_spar z_bot g z_hub
load('model_constants.mat');
%% General Constants
global t Hs Tp rho_air A_r V_rated CT_0 aCT bCT gammaCT
for ig=1:1
    fHighCut = 0.5;          % cut-off frequency
    Tdur = 1600;             % total duration: 1000s transient + 600s
        response
    dtode = 0.1;             % time-step for ode4 solver
    dt = 0.05;               % general time step
    tode4 = [0:dtode:Tdur-dtode];
    t = [0:dt:Tdur-dt];
    z_buoy = z_bot/2;        % center of buoyancy force
    Hs = 6;                  % linear wave amplitude and significant
        wave height [m]
    Tp = 10;                 % linear wave period and significant wave
        period [s]
    rho_air = 1.22;          % air density [kg/m^3]
    A_r = 24885;             % 10 MW rotor area [m^2]
    V_rated = 11.4;          % 10 MW rated wind speed [m/s]
    CT_0 = 0.81;             % 10 MW nominal thrust coefficient
    aCT = 0.5;               % 10 MW "a" thrust parameter
    bCT = 0.65;              % 10 MW "b" thrust paremeter
    gammaCT = 2;             % initial controller value
    gammaJS = 3.3;           % JONSWAP peak enhancement factor
    df = 1/Tdur;             % JONSWAP frequency spectra time step
    TI = 0.14;               % Kaimal wind spectra turbulence intensity
    TL = 340.2;              % Kaimal wind spectra turbulenec length
        scale [m]
end
%% Part 1, Model Formulation
% Question 1 : Mtot, zCMtot and IOtot calculation
for i1=1:1
```

```
    M_tot   = M_floater+ M_tower+M_nacelle+M_rotor;
    zCM_tot=(M_floater*zCM_floater + M_tower*zCM_tower + (M_nacelle+M_rotor)*z_hub)/ M_tot;
    IO_tot = (ICM_floater+ M_floater*zCM_floater^2)+(ICM_tower+M_tower*zCM_tower^2)+(M_nacelle+M_rotor)*z_hub^2;
end
% Question 2,3,4,5 : written in the report
% Question 6 : Building ODE system
for i6=1:1
    I11A = pi/64*D_spar^4;                                              % second moment of area at the waterplane
    M = [M_tot, M_tot*zCM_tot;
         M_tot*zCM_tot, IO_tot];                             % mass matrix
    A = [-pi/4*rhow*D_spar^2*Cm*z_bot, -pi/8*rhow*D_spar^2*Cm*z_bot^2;
         -pi/8*rhow*D_spar^2*Cm*z_bot^2, -pi/12*rhow*D_spar^2*Cm*z_bot^3];    % added mass matrix
    C = [K_moor, K_moor*z_moor;
         K_moor*z_moor, rhow*g*I11A+M_tot*g*(z_buoy-zCM_tot)+K_moor*z_moor^2];
end
% Question 7: Estimating natural frequencies of surge and pitch
global w1 w5
for i7=1:1
    [~,w15]=eig((M+A)^-1*C);
    w1=w15(1,1)^0.5/(2*pi);                 % Hz
    w5=w15(2,2)^0.5/(2*pi);                 % Hz
end
%% Part 2, Dynamic Analysis
% Question 8: written in the report
% Question 9: preparing for ode4 solver
for i9=1:1
    B = [2*10^5, 0; 0, 0];                                 % Damping matrix
end
% Question 10: implementing ode4/dqdt algorithm
for i10=1:1
    q01 = [1;0;0;0;0];                                     % first initial condition
    q02 = [0;0.1;0;0;0];                                   % second initial condition
    Y10_1 = ode4(@dqdtsparbuoy,tode4,q01,M+A,B,C,1);     % unforced response 1
    Y10_2 = ode4(@dqdtsparbuoy,tode4,q02,M+A,B,C,1);     % unforced response 2
    Y10_1(:,2)=Y10_1(:,2)*180/pi;                         % convert pitch
```

```
to degrees
    Y10_2(:,2)=Y10_2(:,2)*180/pi;
    % plotting PSD
    labels10=["Surge, x_0 [m]","Pitch, \theta [deg]";"PSD [m^2/Hz]","PSD [deg^2/Hz]"];
    figure
    PSD(tode4,Y10_1(:,1:2),fHighCut,labels10)
    figure
    PSD(tode4,Y10_2(:,1:2),fHighCut,labels10)
end
% Question 11: with hyrdodynamic forcing, assuming no current u = 0
global u udot z
for i11=1:1
    % part a
    z = [0:-1:z_bot];
    u=zeros(length(z),length(t));
    udot=u;
    Y11_1 = ode4(@dqdtsparbuoy,tode4,q01,M+A,B,C,2);%hydro forcing response 1
    Y11_2 = ode4(@dqdtsparbuoy,tode4,q02,M+A,B,C,2);%hydro forcing response 2
    Y11_1(:,2)=Y11_1(:,2)*180/pi;                    % convert pitch to degrees
    Y11_2(:,2)=Y11_2(:,2)*180/pi;
    % plotting PSD
    figure
    PSD(tode4,Y11_1(:,1:2),fHighCut,labels10)
    figure
    PSD(tode4,Y11_2(:,1:2),fHighCut,labels10)
    % part b
    q03 = [0;1;0;0;0];
    Y11_3 = ode4(@dqdtsparbuoy,tode4,q03,M+A,B,C,1);%no forcing response
    Y11_4 = ode4(@dqdtsparbuoy,tode4,q03,M+A,B,C,2);%hydro forcing response
    Y11_3(:,2)=Y11_3(:,2)*180/pi;                    % convert pitch to degrees
    Y11_4(:,2)=Y11_4(:,2)*180/pi;                    % convert pitch to degrees
    % plotting PSD
    figure
    PSD(tode4,Y11_3(:,1:2),fHighCut,labels10)
    PSD(tode4,Y11_4(:,1:2),fHighCut,labels10)
end
% Question 12: Linear Wave Sea-State Hydrodynamic forcing
for i12=1:1
```

```
    [u, udot]=LinearWaveKinematics();
    q00 = [0;0;0;0;0];                                      % zero initial condition
    Y12 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,2);  % hydro forcing response 0
    Y12(:,2)=Y12(:,2)*180/pi;                              % convert pitch to degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y12(10001:16000,1:2),fHighCut,labels10)          % make sure to eliminate transient
%     ylim([-1,1])
end
% Question 13: V10 Wind and Linear Wave Forcing
global V_10 V_hub
for i13=1:1
    V_10 = 8;
    V_hub = 8*ones(size(t));
    Y13 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,3);  % hydro forcing response 0
    Y13(:,2)=Y13(:,2)*180/pi;                              % convert pitch to degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y13(10001:16000,1:2),fHighCut,labels10)          % make sure to eliminate transient
end
% Question 14: V10 Wind and Jonswap Wave Forcing
for i14=1:1
    [fvec,amp,S_JS] = jonswap(Hs,Tp,df,fHighCut,gammaJS);
    [u,udot]=IrregularWaveKinematics(fvec,amp) ;
    Y14 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,3);  % hydro forcing response 0
    Y14(:,2) = Y14(:,2)*180/pi;                            % convert pitch to degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y14(10001:16000,1:2),fHighCut,labels10)          % make sure to eliminate transient
end
% Question 15: Kaimal Wind and Jonswap Wave Forcing
for i15=1:1
    [~,V_hub,~] = Kaimal_Timeseries(TI,TL,fHighCut);
    Y15 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,3);  % hydro forcing response 0
    Y15(:,2) = Y15(:,2)*180/pi;                            % convert pitch to
```

```
        degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y15(10001:16000,1:2),fHighCut,labels10)
            % make sure to eliminate transient
end
%% Part 3: Adaptation of pitch control for dynamic stability
% Adapting the CT pitch controller for a floating configuration
% Question 16: Steady wind and no waves
for i16=1:1
    % 10 m/s case no waves
    V_10 = 10;
    V_hub = 10*ones(size(t));
    u=zeros(length(z),length(t));
    udot=u;
    Y16_1 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,3);  % hydro forcing
        response 0
    Y16_1(:,2)=Y16_1(:,2)*180/pi;                      % convert pitch
         to degrees
    % 16 m/s case no waves
    V_10 = 16;
    V_hub = 16*ones(size(t));
    Y16_2 = ode4(@dqdtsparbuoy,tode4,q00,M+A,B,C,3);  % hydro forcing
        response 0
    Y16_2(:,2)=Y16_2(:,2)*180/pi;                      % convert pitch
         to degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y16_1(10001:16000,1:2),fHighCut,labels10)
            % make sure to eliminate transient
    figure
    PSD(tode4(10001:16000),Y16_2(10001:16000,1:2),fHighCut,labels10)
            % make sure to eliminate transient
end
% Question 17: CT adapted controller model
for i17=1:1
    % 16 m/s case no waves
    q00contr = [0;0;0;0;0];
    Y17 = ode4(@dqdtsparbuoy,tode4,q00contr,M+A,B,C,4);  % hydro
        forcing response 0
    Y17(:,2) = Y17(:,2)*180/pi;                        % convert pitch
        to degrees
    % plotting PSD
    figure
    PSD(tode4(10001:16000),Y17(10001:16000,1:2),fHighCut,labels10)
            % make sure to eliminate transient
```

```
end
% Question 18: Proper gamma for CT controller model
for i18=1:1
    gammaCTrange = [0.1,0.5,1.5];
    Y18 = zeros(length(Y17),length(gammaCTrange));
    figure
    for igam = 1:length(gammaCTrange)
        gammaCT = gammaCTrange(igam);
        Y18i = ode4(@dqdtsparbuoy,tode4,q00contr,M+A,B,C,4);      % hydro forcing response 0
%           Y18(:,igam) = Y18i(:,2);                              % looking only at surge
        Y18i(:,2) = Y18i(:,2)*180/pi;                              % convert pitch to degrees
        PSD(tode4,Y18i(:,1:2),fHighCut,labels10)
        hold on
    end
end
```

Functions, alphabetically listed

```
function dq = dqdtsparbuoy(tode4,q,M,B,C,forcing)
%% Description
% This function takes a mass matrix, M (add added mass before), damping
% matrix, B, restoring matrix, C, and initial
% conditions, q
%
% The function then calculates the relevant forcing vector, depending on
% the forcing input case, forcing
%
% The function then returns a generalized motion, dq
%
% This function is developed for solving the surge and pitch motions of a
% floating, moored spar-buoy wind turbine system
%
% Future development will include making this function robust for various
% degree-of-freedom considerations
%
% As it is currently implemented, M,B,C are all 2x2 matrices; F is a 2x1
% forcing vector with Force in 1 and torque in 2; q is a 4x1 vector that is
% split into two 2x1 vectors for solving the system. The return, dq, is 4x1
```

```
%% Implementation
% Find the proper time for forcing
global t z_hub gammaCT
[~ ,index]= min(abs(tode4-t));
% Dummy CT_vrel if not using controller
CT_vrel = 0;
% Calculate forcing
if forcing == 1
    F = [0;0];                                                          % no external forcing
elseif forcing == 2
    F = hydroforcing(index,q(3),q(4));                          % hydro forcing only
elseif forcing == 3
    q3hub = q(3)+z_hub*q(4);                                    % dxdt of the hub
    [Windforcing,~] = F_wind_timepoint(q3hub,index);
    F = hydroforcing(index,q(3),q(4)) + Windforcing;          % hydro plus steady wind forcing
elseif forcing == 4
    q3hub = q(3)+z_hub*q(4);                                    % dxdt of the hub
    [Windforcing,CT_vrel] = F_wind_controller(q3hub,q(5),index);     % time-delay controller
    F = hydroforcing(index,q(3),q(4)) + Windforcing;          % hydro plus steady wind forcing
end
dq34 = (M)^-1*(F-B*q(3:4)-C*q(1:2));         % calculate qdot 3 and qdot 4
dq5 = -gammaCT*(q(5)-CT_vrel);
dq=[q(3);q(4);dq34(1);dq34(2);dq5];              % return qdot vector 4x1
```

```
function [F_Tauwind,CT_vrel] = F_wind_controller(dxdt_hub,q5,tindex)
%% Description
% This function calculates the forcing from wind, and can compute for
% steady wind or for unsteady time series.
% This function takes the V_10 time average, V_hub, a and b parameters, and
% the hub motion to calculate the relative velocity, corrected CT, and
% determine thrust.
% This function returns the forcing and torque for a given point in time.
%% Implementation
global rho_air A_r V_rated CT_0 V_10 V_hub aCT bCT z_hub
% Find V_rel
```

```
V_rel = V_hub(tindex)-dxdt_hub;
% Set values for CT_hub
if V_rel <= V_rated
    CT_hub = CT_0;
else
    CT_vrel = CT_0 *exp(-aCT*(V_rel-V_rated)^bCT);
    CT_hub = q5;
end
% Set values for CT_10
if V_10 <= V_rated
    CT_10 = CT_0;
else
    CT_10 = CT_0 *exp(-aCT*(V_10-V_rated)^bCT);
end
% Calculate mean aero force
Fwind_m = 0.5*rho_air*A_r*CT_10*V_10^2;
% Calculate time-varying aero force
Fwind_t = 0.5*rho_air*A_r*CT_hub*V_rel^2;              % using the CT
    calculated previously using the update
% Calculate reduction factor
if V_10<V_rated
    f_red = 0.54;
elseif V_10
    f_red = 0.54 + 0.027*(V_10-V_rated);
end
Fwind = Fwind_m + f_red*(Fwind_t-Fwind_m);
Tauwind = Fwind*z_hub;
F_Tauwind=[Fwind;Tauwind];
```

```
function [F_Tauwind,CT_hub] = F_wind_timepoint(dxdt_hub,tindex)
%% Description
% This function calculates the forcing from wind, and can compute for
% steady wind or for unsteady time series.
% This function takes the V_10 time average, V_hub, a and b parameters,
    and
% the hub motion to calculate the relative velocity, corrected CT, and
% determine thrust.
% This function returns the forcing and torque for a given point in
    time.
%% Implementation
global rho_air A_r V_rated CT_0 V_10 V_hub aCT bCT z_hub
% Find V_rel
V_rel = V_hub(tindex)-dxdt_hub;
% Set values for CT_hub
if V_rel <= V_rated
    CT_hub = CT_0;
else
```

```matlab
    CT_hub = CT_0 *exp(-aCT*(V_rel-V_rated)^bCT);
end
% Set values for CT_10
if V_10 <= V_rated
    CT_10 = CT_0;
else
    CT_10 = CT_0 *exp(-aCT*(V_10-V_rated)^bCT);
end
% Calculate mean aero force
Fwind_m = 0.5*rho_air*A_r*CT_10*V_10^2;
% Calculate time-varying aero force
Fwind_t = 0.5*rho_air*A_r*CT_hub*V_rel^2;
% Calculate reduction factor
if V_10<V_rated
    f_red = 0.54;
elseif V_10
    f_red = 0.54 + 0.027*(V_10-V_rated);
end
Fwind = Fwind_m + f_red*(Fwind_t-Fwind_m);
Tauwind = Fwind*z_hub;
F_Tauwind=[Fwind;Tauwind];
```

```matlab
function Fvec = hydroforcing(t_index,q3,q4)
%% Description
% This function can evaluate the forces and moments from hydrodynamic
% forcing for still water or for a given sea state. This function is
% designed for use with dqdtsparbuoy and ode4.
%% Inputs:
% t is the current time state and will be used to pull the pre-determined
% sea state
% q3 is dx0/dt
% q4 is d_theta/dt
%% Outputs:
% Hydrodynamic force and moment for a point in time
%% Implementation
global rhow Cm CD D_spar u udot z
A = D_spar^2*pi/4;
F = 0;
Tau = 0;
for i=1:length(z)
    df = rhow*((Cm+1)*A*udot(i,t_index) + 0.5*CD*D_spar*(u(i,t_index)-q3-z(i)*q4)*abs(u(i,t_index)-q3-z(i)*q4));
    dtau = df*z(i);
    F = F + df;
    Tau = Tau + dtau;
end
```

```
Fvec=[F;Tau];
```

```
function [u,a]=IrregularWaveKinematics(fvec,amp)
% This function calculates the velocity and acceleration at various heights
% for a floating spar for irregular waves
% Inputs
% fvec = wave frequency
% h = depth of water or spar
% g = gravity
% amp = wave amplitude
% rho = density of water
% U= Horizontal Velocity
global g z_bot z t
% Given constants
for gc = 1:1
    h = -z_bot;
end
% Calculated Constants and Initializations
for cc = 1:1
    k = zeros(size(fvec));
    x = 0;
    u = zeros(length(z),length(t));
    a = zeros(length(z),length(t));
end
% Frequency domain inputs for wave number
for ifreq = 1:length(fvec)
    k(ifreq) = wave_number(fvec(ifreq),g,h);
end
% random error
random = 2*pi*rand(1,length(fvec));
% calculating acceleration and velocity
for iz=1:length(z)
    for it=1:length(t)
        uj = 0;
        aj = 0;
        for ifreq = 2:length(fvec)
            omega = 2*pi*fvec(ifreq);
            uj = uj + amp(ifreq) *omega* cosh(k(ifreq)*(z(iz)+h)) / sinh(k(ifreq)*h)*cos(omega*t(it)-(k(ifreq)*x)+(random(ifreq)));
            aj = aj - omega^2*amp(ifreq) * cosh(k(ifreq)*(z(iz)+h)) / sinh(k(ifreq)*h) * sin(omega*t(it)-(k(ifreq)*x)+(random(ifreq)));
        end
        u(iz,it) = uj;
        a(iz,it) = aj;
```

```
    end
end
```

```
function [fvec,a,S_JS] = jonswap(Hs,Tp,df,fHighCut,gammaJS)
    % This function calculates the JONSWAP distribution for waves,
    % frequency and amplitude
    % Inputs: Hs, Period, frequency step, max frequency considered, gamma
    % Outputs: time-varying frequency, time-varying wave ampltiude, Jonswap
    % frequency spectra
    fvec = [0 : df : fHighCut];
    fp= 1/Tp;
    for i =1: length(fvec)
        if fvec(i) <= fp
            sigma = 0.07;
        else
            sigma = 0.09;
        end
        gammaexp = exp(-0.5*(((fvec(i)/fp)-1)/sigma)^2);
        S_JS(i) = 0.3125* Hs^2 *Tp * (fvec(i)/fp)^(-5)* exp(-1.25*(fvec(i)/fp)^(-4))*(1-0.287*log(gammaJS))*gammaJS^gammaexp;
        a(i) = sqrt(2*S_JS(i)*df);
    end
return
```

```
function [S_W,V,f] = Kaimal_Timeseries(I,L,fHigh)
%% Description
% This function calculates wind power density using the Kaimal spectrum. It
% takes wind parameter inputs:
% I: turbulence intensity
% V_10: 10-minute average wind speed [m/s]
% L: turbulence length scale [m]
% fHigh: cut-off frequency
% t: time space [s]
% and returns:
% the spectral density function, S_W
% the velocity time series, V, and
% the freqeuency vector, f.
%% Implementation
% initializations
global t V_10
df = 1/t(end);
f = [df:df:fHigh];
rng(1)
ep = 2*pi*rand(1,length(f));
```

```
bp = zeros(1,length(f));
wp = 2*pi*f;
S_W = zeros(size(t));
V = zeros(size(t));
% calculate spectrum
for p = 1: length(f)
    S_W(p) = 4*I^2*V_10*L/(1+6*(f(p)*L/V_10))^(5/3);
    bp(p) = sqrt(2*S_W(p)*df);
end
% velociy time series
for i = 1:length(t)
    V(i) = V_10 + sum(bp.*cos(wp*t(i)+ep));
end
```

```
function [u,a]=LinearWaveKinematics()
% This function calculates the kinematics of regular waves
% Inputs
% f= wave frequency
% h= depth of water or spar
% g= gravity
% rho = density of water
% U= Horizontal Velocity
global g z_bot Hs Tp z t
% Given constants
    h = -z_bot;
    f = 1/Tp;
    H = Hs;
% Calculated Constants
    w=2*pi*f;
    k=wave_number(f,g,h);
    % Pre calculations
    x3=0;
    u=zeros(length(z),length(t));
    a=zeros(length(z),length(t));
    for j=1:length(z)
        for i=1:length(t)
            u(j,i) = w*H/2 * cosh(k*(z(j)+h)) / sinh(k*h) * cos(w*t(i)-
                k*x3);
            a(j,i) = -w^2*H/2 * cosh(k*(z(j)+h)) / sinh(k*h) * sin(w*t(
                i)-k*x3);
        end
    end
```

```
function Y = ode4(odefun,tspan,y0,varargin)
%ODE4  Solve differential equations with a non-adaptive method of order
    4.
%   Y = ODE4(ODEFUN,TSPAN,Y0) with TSPAN = [T1, T2, T3, ... TN]
```

```
integrates
%   the system of differential equations y' = f(t,y) by stepping from T0 to
%   T1 to TN. Function ODEFUN(T,Y) must return f(t,y) in a column vector.
%   The vector Y0 is the initial conditions at T0. Each row in the solution
%   array Y corresponds to a time specified in TSPAN.
%
%   Y = ODE4(ODEFUN,TSPAN,Y0,P1,P2...) passes the additional parameters
%   P1,P2... to the derivative function as ODEFUN(T,Y,P1,P2...).
%
%   This is a non-adaptive solver. The step sequence is determined by TSPAN
%   but the derivative function ODEFUN is evaluated multiple times per step.
%   The solver implements the classical Runge-Kutta method of order 4.
%
%   Example
%         tspan = 0:0.1:20;
%         y = ode4(@vdp1,tspan,[2 0]);
%         plot(tspan,y(:,1));
%     solves the system y' = vdp1(t,y) with a constant step size of 0.1,
%     and plots the first component of the solution.
%

if ~isnumeric(tspan)
  error('TSPAN should be a vector of integration steps.');
end

if ~isnumeric(y0)
  error('Y0 should be a vector of initial conditions.');
end

h = diff(tspan);
if any(sign(h(1))*h <= 0)
  error('Entries of TSPAN are not in order.')
end

try
  f0 = feval(odefun,tspan(1),y0,varargin{:});
catch
  msg = ['Unable to evaluate the ODEFUN at t0,y0. ',lasterr];
  error(msg);
end
```

```
y0 = y0(:);    % Make a column vector.
if ~isequal(size(y0),size(f0))
  error('Inconsistent sizes of Y0 and f(t0,y0).');
end

neq = length(y0);
N = length(tspan);
Y = zeros(neq,N);
F = zeros(neq,4);

Y(:,1) = y0;
for i = 2:N
  ti = tspan(i-1);
  hi = h(i-1);
  yi = Y(:,i-1);
  F(:,1) = feval(odefun,ti,yi,varargin{:});
  F(:,2) = feval(odefun,ti+0.5*hi,yi+0.5*hi*F(:,1),varargin{:});
  F(:,3) = feval(odefun,ti+0.5*hi,yi+0.5*hi*F(:,2),varargin{:});
  F(:,4) = feval(odefun,tspan(i),yi+hi*F(:,3),varargin{:});
  Y(:,i) = yi + (hi/6)*(F(:,1) + 2*F(:,2) + 2*F(:,3) + F(:,4));
end
Y = Y.';
```

```
function [] = PSD(t,signal,fHighCut,ylabelstr)
%% Description
% This function takes a timeseries (t) and the signal response for that
% time series as an input, and return a plot of the time  series and
% frequency domain. It also takes the cutoff frequency for plotting.
% The dimension of the signal tells the function how many subplots to make.
% The user also designates the y-axis labels for the signal being plotted
% with a matrix of strings, size 2 x (number of signals).
    %% Important PSD information for plotting only steady-state response
    % when plotting time decay, set timestartpos to 1. when plotting
    % forced response, set timestartpos to 10001
    timestartpos = 1;       % steady state time position 1
%% Implementation
% get the number of signals to be plotted, and the relevant direction of
% the signal input matrix
global w1 w5
[numbersignals,mindim] = min(size(signal));
% if needed, transpose the signal matrix
if mindim==1
```

```
    signal=signal';
end
% create subplots
for numbersubplots=1:numbersignals
    % Plot timeseries
    subplot(numbersignals,2,2*numbersubplots-1), plot(t,signal(:,
       numbersubplots),'LineWidth',1.25), grid on
    hold on
    % xlabel only if the last plot
    if numbersubplots==numbersignals
        xlabel('Time [s]')
    end
    ylabel(ylabelstr(1,numbersubplots))

    df = 1/(t(end)-t(timestartpos));          % Frequency resolution
    fpsd = df*(0:length(t)-timestartpos);  % Frequency vector starts
       from 0 for length t

    signalhat = fft(signal(timestartpos:end,numbersubplots))/length(t(
       timestartpos:end));              % Fourier amplitudes
    signalhat(1) = 0;                            % Discard first value (
       mean)
    signalhat(round(length(fpsd)/2):end) = 0;    % Discard all above
       Nyquist fr.
    signalhat = 2*signalhat;                     % Make amplitude one-
       sided
    psd = abs(signalhat).^2/2/df;                % Calculate spectrum

    % Plot frequency domain
    subplot(numbersignals,2,2*numbersubplots), plot(fpsd,psd,'LineWidth
       ',1.25),
    xline(w1,'--r')
    xline(w5,'--m')
    grid on
    hold on, xlim([0 0.12])
    legend('','f_{s}=0.0083 Hz','f_{p}=0.0326 Hz')
    if numbersubplots==numbersignals
        xlabel('Frequency [Hz]')
    end
    ylabel(ylabelstr(2,numbersubplots))
end
```

```
function k = wave_number(f,g,h)
% this function calculates the wave number from frequency (f) and depth
   (h)
% w = radian frequency
%
```

```
% w=2*pi*f;

fun = @(k) (2*pi*f)^2 - g*k*tanh(k*h);

k=fzero(fun,[0 3]);
```