

Linear Quadratic Control Design

The Linear Quadratic Regulator (LQR) and the Linear Quadratic Gaussian (LQG) control design are easy to use methods for designing controls to stabilize and regulate systems. The LQR is simply state-feedback. The LQG is used when the plant states are not directly available for measurement. It consists of two steps: the design of an LQR state-feedback controller, and the design of a Kalman-Filter observer in order to estimate the state vector. The state-feedback and the estimator are combined together to create an output feedback dynamic controller in state-space form.

1. Linear Quadratic Regulator

The Linear Quadratic Regulator (LQR) is used to design state-feedback control gains that stabilize a plant model, achieve good closed-loop performance of the states in response to transients and robustness to parameter uncertainties. It requires a plant model in state-space form. The plant inputs are the controls and the outputs are either measurements or criteria to be optimized. The plant must be stabilizable from the controls and detectable from the outputs. The control solution is a feedback from the plant states derived by the optimization of a linear quadratic performance index using the Riccati equation. The optimization takes into consideration two important and most frequently conflicting requirements: the speed of convergence of the state-vector from some initial value and the amount of the control input along a trajectory. We will present the analytic solutions for both the continuous and discrete LQR problems.

1.1 The Continuous Asymptotic LQR Problem

Equation 1.1.1 represents the plant dynamics in state-space form

$$\begin{aligned}\dot{\underline{x}}(t) &= A \underline{x}(t) + B \underline{u}(t) \\ \underline{y}(t) &= C \underline{x}(t)\end{aligned}\tag{1.1.1}$$

Where:

$\underline{x}(t)$ is the state of dimension n
 $\underline{u}(t)$ is the control of dimension m
 $\underline{y}(t)$ is the output of dimension r

The LQR method calculates a state-feedback optimal control $\underline{u}^o(t)$ that minimizes the quadratic performance index J in equation 1.1.3.

$$\begin{aligned}\underline{u}^o(k) &= -K_c \underline{x}(k) \\ J &= \int_0^{\infty} \left[\underline{y}(t)^T Q \underline{y}(t) + \underline{u}(t)^T R \underline{u}(t) \right] dt\end{aligned}\tag{1.1.2, 3}$$

Where: the matrices Q and R, are defined to be the output and control weight matrices. They are used as knobs which adjust the closed-loop system's response to initial states \underline{x}_0 or to disturbances. They trade performance in the output vector $\underline{y}(t)$ in terms of speed of convergence versus the magnitudes of control $\underline{u}(t)$. The (r x r) matrix Q should be symmetric, positive semidefinite, and the matrix R (m x m) should be symmetric positive definite.

Selecting a small R or a large Q in the LQR design, it is telling the mathematics that when the control loop is closed and the system is initialized at some initial state $\underline{x}(0)$, I would like the output response $\underline{y}(t)$ to convergence fast to the commanded value with small transients, regardless of how much control force $\underline{u}(t)$ is necessary to achieve this. This results in a high bandwidth control system and possibly effector saturation.

If on the other hand a large R or small Q are used in equation 1.1.3, the magnitudes of the control $\underline{u}(t)$ are penalized more than the output transients $\underline{y}(t)$ in the performance index. It indicates that my actuators do not have as much strength to handle disturbances or big commands. The closed-loop system's response to disturbances will be slower, resulting in a reduced control system bandwidth that will protect the actuators from saturating.

The solution to the above problem exists if the (A, B) system is stabilizable, and the (A, D) pair is detectable where D is defined by equation 1.1.4. it means that all unstable plant modes must be controllable and measurable.

$$D^T D = C^T Q C \quad (1.1.4)$$

Notice, that the variable $\underline{y}(t)$ used in the optimization criterion is not necessarily the plant output. Any combination of output variables, not necessarily measurable, represented by a matrix C_1 , different than C, can be used in the optimization criterion, as long as the plant states are detectable from C_1 . C_1 may also be the identity matrix, in which case the state variables are directly and individually penalized in the performance index via matrix Q.

The state-feedback gain K_c of equation 1.1.2 is calculated from equation 1.1.5, and the (n x n) matrix P is obtained from the steady-state solution of the Riccati Equation 1.1.6

$$K_c = R^{-1} B^T P \quad \text{where:} \quad 1.1.5, 6$$

$$-\dot{P} = P A + A^T P + C^T Q C - P B R^{-1} B^T P = 0$$

Furthermore if the problem is initialized with an initial state error $\underline{x}(0)=\underline{x}_0$, then the performance index criterion is: $J = \underline{x}(0)^T P \underline{x}(0)$

1.2 The Discrete Asymptotic Case

In the discrete case the plant system is represented by the difference matrix equations 1.2.1.

$$\begin{aligned} x(k+1) &= A x(k) + B u(k) \\ y(k) &= C x(k) \end{aligned} \tag{1.2.1}$$

Where:

$\underline{x}(k)$ is the state of dimension n

$\underline{u}(k)$ is the control of dimension m

$\underline{y}(k)$ is the output of dimension r

k represents the present state variable and (k+1) is the next iteration

The Discrete LQR method calculates a state-feedback optimal control $u^o(k)$ that minimizes the quadratic performance index J in equation 1.2.2.

$$\begin{aligned} \underline{u}^o(k) &= -K_c \underline{x}(k) \\ J &= \lim(N \rightarrow \infty) \sum_0^{N-1} [y(k+1)^T Q y(k+1) + u(k)^T R u(k)] \end{aligned} \tag{1.2.2-3}$$

Where: the matrices Q and R, are defined as in the continuous case. The (m x n) state feedback gain matrix K_c is obtained from equation 1.2.4, where: the (n x n) matrix P is obtained from the solution of the discrete steady-state Riccati Equation 1.2.5

$$\begin{aligned} K_c &= (R + B^T P_{k+1} B)^{-1} B^T P_{k+1} A \\ P_k &= [A - B K_c]^T P_{k+1} [A - B K_c] + K_c^T R K_c + C^T Q C = 0 \end{aligned} \tag{1.2.4-5}$$

Where: k= 0, 1, 2, 3, ..., N-1

The solution to the above problem exists if the (A, B) system is stabilizable, and the (A, D) pair is detectable, where D is defined by equation 1.2.6. This means that all unstable plant modes must be controllable and measurable.

$$D^T D = C^T Q C \tag{1.2.6}$$

Furthermore if the problem is initialized with an initial state error $\underline{x}(0)=\underline{x}_0$, then the performance index criterion is: $J = \underline{x}(0)^T P(0) \underline{x}(0)$

2. The Finite-Time LQR with Terminal State Penalty

The finite-time or transient deterministic optimal linear quadratic regulator problem is essentially similar to the steady-state case described in section 1. The difference is that an additional term is included in the performance index that penalizes the value of the state-vector $\underline{x}(t_f)$ at the terminal time t_f . The resulting state-feedback control law $K_c(t)$ is time-varying.

2.1 The Continuous Transient LQR Problem

The plant dynamics in state-space form is the same as before and the system is initialized at $\underline{x}(0)=\underline{x}_0$

$$\begin{aligned}\dot{\underline{x}}(t) &= A\underline{x}(t) + B\underline{u}(t) + w(t) \\ \underline{y}(t) &= C\underline{x}(t)\end{aligned}\tag{2.1.1}$$

Where:

- $\underline{x}(t)$ is the state of dimension n
- $\underline{u}(t)$ is the control of dimension m
- $\underline{y}(t)$ is the output of dimension r
- $w(t)$ is white noise with intensity $V(t)$

The Transient LQR solution calculates a state-feedback optimal control $\underline{u}^0(t)$ that minimizes the quadratic performance index J in equation 2.1.3.

$$\begin{aligned}\underline{u}^0(t) &= -K_c(t)\underline{x}(t) \text{ where:} \\ J &= \int_0^{t_f} \left[\underline{y}(t)^T Q \underline{y}(t) + \underline{u}(t)^T R \underline{u}(t) \right] dt + \underline{x}(t_f)^T P_1 \underline{x}(t_f)\end{aligned}\tag{2.1.2-3}$$

Where: (t_f) is a known terminal time. The matrices Q and R are the output and control weight matrices as already described in Section 1. P_1 is a weight matrix that penalizes the terminal state. These matrices determine the optimal trade-off between: the output $\underline{y}(t)$ deviations from zero along the trajectory, the magnitude of control input $\underline{u}(t)$, and the dispersion of the terminal state vector $\underline{x}(t_f)$ from zero at the end-time.

Matrix Q is $(r \times r)$ and should be symmetric positive semidefinite,
Matrix R is $(m \times m)$ and should be symmetric positive definite,
Matrix P_1 is $(n \times n)$ and should be symmetric positive semidefinite

The (m x n) state feedback gain matrix $K_c(t)$ is obtained from equation 2.1.4, where: the (n x n) matrix $P(t)$ is obtained from the solution of the transient Riccati Equation 2.1.5

$$K_c(t) = R^{-1} B^T P(t) \text{ where:} \quad (2.1.4, 5)$$

$$\dot{P} = P A + A^T P + C^T Q C - P B R^{-1} B^T P$$

The equation 2.1.5 is solved backwards in time after being initialized at the terminal time t_f where: $P(t_f) = P_1$

Furthermore the performance index criterion J is

$$J = \underline{x}(0)^T P \underline{x}(0) + \int_0^{t_f} \text{trace}[P(t)V(t)] dt \quad (2.1.6)$$

Solution of the Continuous Transient LQR

The following algorithm, from reference [1], is used to solve the transient regulator problem. Let us define a matrix Z where:

$$Z = \begin{bmatrix} A & -B R^{-1} B^T \\ -C^T Q C & -A \end{bmatrix} \quad (2.1.7)$$

Matrix Z has the property that if α is an eigenvalue of Z , $-\alpha$ is also an eigenvalue of Z . We can find the eigenvector matrix W such that

$$Z = W \begin{bmatrix} \Lambda & 0 \\ 0 & -\Lambda \end{bmatrix} W^{-1} \quad (2.1.8)$$

Where: Λ is a diagonal matrix consisting of the positive eigenvalues of Z , and $-\Lambda$ consisting of the negative eigenvalues of Z . Then we partition the (2n x 2n) matrix W into four (n x n) blocks as shown in equation 2.1.9

$$W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix} \quad (2.1.9)$$

The solution $P(t)$ of the Riccati equation 2.1.5 is obtained from equation 2.1.10, where

$$P(t) = [W_{22} + W_{21} G(t_f - t)][W_{12} + W_{11} G(t_f - t)]^{-1} \text{ where:} \quad (2.1.10-12)$$

$$G(t) = \exp(\Lambda t) S \exp(-\Lambda t), \text{ and}$$

$$S = -(W_{22} - P_1 W_{12})^{-1} (W_{21} - P_1 W_{11})$$

From equation (2.1.11) as (t_f) approaches infinity, $G(t_f - t)$ approaches zero, and the steady-state solution of the Riccati equation becomes: $P = W_{22} W_{12}^{-1}$

2.2 The Discrete Time Transient LQR

In the Discrete Transient LQR design case the plant system is represented by the difference matrix equations 2.2.1

$$\begin{aligned} \underline{x}(k+1) &= A \underline{x}(k) + B \underline{u}(k) + w(k) \\ \underline{y}(k) &= C \underline{x}(k) \end{aligned} \quad (2.2.1)$$

Where:

$\underline{x}(k)$ is the state of dimension n
 $\underline{u}(k)$ is the control of dimension m
 $\underline{y}(k)$ is the output of dimension r
 $w(k)$ is zero mean white noise with variance V(t)

We must calculate a state-feedback optimal control $\underline{u}^o(k)$ that minimizes the quadratic performance index J in equation 2.2.3.

$$\begin{aligned} \underline{u}^o(k) &= -K_c(k) \underline{x}(k) \\ J &= \lim(N \rightarrow \infty) \sum_0^{N-1} \left[\underline{y}(k+1)^T Q \underline{y}(k+1) + \underline{u}(k)^T R \underline{u}(k) \right] + \underline{x}(N)^T P_N \underline{x}(N) \end{aligned} \quad (2.2.2, 3)$$

The matrices Q and R and P_N are defined as in the continuous case. The discrete optimal regulator solution is obtained from the difference equation 2.2.4 solved backwards in time, initialized at N with $P(N)=P_N$. The (n x n) matrix P(k) is obtained from the solution of the discrete transient Riccati Equation 2.2.5, where: k=0, 1, 2,..., N-1

$$\begin{aligned} K_c(k) &= \left(R + B^T P(k+1) B \right)^{-1} B^T P(k+1) A \\ P(k) &= \left[A - B K_c(k) \right]^{-1} P(k+1) \left[A - B K_c(k) \right] + K_c^T(k) R K_c(k) + C^T Q C = 0 \end{aligned} \quad (2.2.4, 5)$$

Furthermore if the problem is initialized with an initial state error $\underline{x}(0)=\underline{x}_0$, then the performance index criterion is

$$J = \underline{x}(0)^T P(0) \underline{x}(0) + \sum_{j=0}^{j=N-1} \text{trace} \left[V(j) P(j+1) \right] \quad (2.2.6)$$

3. The Asymptotic Kalman-Bucy State-Estimator, Observer

In the previous two sections we demonstrated how to design optimal state-feedback controllers, assuming that the state vector can be measured accurately and be available for feedback. This assumption is most often unrealistic because in most systems the state vector is not directly measurable but the output measurements consist of a linear combination of the states. We therefore need to design an observer, which is a system that will approximately reconstruct the state vector from the plant output, and this will allow us to apply our optimal state-feedback control laws. In this section we will present the design of the steady-state Kalman-Bucy filter that is used to reconstruct an approximation of the state vector from the measured system output that will converge to the state vector. We shall also assume that the system is corrupted by two types of noises: measurement, and state excitation noise. They are both white, zero mean and are not correlated. We will first analyze the continuous and then the discrete Kalman-Filter observer for continuous and discrete systems.

3.1 The Continuous Kalman-Bucy Filter

Let us consider the following plant in state-space form. This system is affected by disturbances $\underline{w}(t)$ and the observations $\underline{y}(t)$ are corrupted by noise $\underline{v}(t)$

$$\begin{aligned}\dot{\underline{x}}(t) &= A\underline{x}(t) + B\underline{u}(t) + Gw(t) \\ \underline{y}(t) &= C\underline{x}(t) + v(t)\end{aligned}\tag{3.1.1}$$

Where:

$\underline{x}(t)$ is the state vector of dimension n .

$\underline{u}(t)$ is the control input vector of dimension m .

$w(t)$ is the process noise of dimension l ($l \leq n$) and covariance matrix Q_{pn} , where $Q_{pn} = Q'_{pn} \geq 0$.

$\underline{y}(t)$ is the measurement vector of dimension r ($r \leq n$).

$\underline{v}(t)$ is the measurement noise with intensity $R_{mn} = R'_{mn} \geq 0$.

The purpose of the optimal Kalman Filter estimator is to construct an estimate of the state x operating over the time range $[t_0 \rightarrow t]$ such that the index J in equation 3.1.2 is minimized, where: $\hat{x}(t)$ denotes the estimate of $x(t)$. E is the expected value, and the matrix W is $(n \times n)$ positive semi-definite.

$$J = \lim(t_0 \rightarrow \infty) E \left[(\underline{x}(t) - \hat{\underline{x}}(t))^T W (\underline{x}(t) - \hat{\underline{x}}(t)) \right]\tag{3.1.2}$$

The solution exists when the pair (A^T, C^T) is stabilizable and the pair (A, D) is detectable

Where: $D D' = G Q_{pn} G^T$

After initializing with the expected value of the initial state: $\hat{\underline{x}}(0) = E[\underline{x}(0)]$, the state estimate is computed by the following differential equation 3.1.4. This equation can also be written as in 3.1.5 to show that the inputs to the estimator are the plant control vector $\underline{u}(t)$ and the measurements $\underline{y}(t)$, as shown in figure (3.1)

$$\begin{aligned} \dot{\hat{\underline{x}}}(t) &= A \hat{\underline{x}}(t) + B \underline{u}(t) + K_f [y(t) - C \hat{\underline{x}}(t)] \\ \dot{\hat{\underline{x}}}(t) &= [A - K_f C] \hat{\underline{x}}(t) + B \underline{u}(t) + K_f y(t) \end{aligned} \quad (3.1.4, 5)$$

The steady-state Kalman-Filter gain K_f is obtained from equation 3.1.6, where matrix P is symmetric positive semi-definite and is obtained from the steady-state solution of the Asymptotic Riccati Equation 3.1.7

$$\begin{aligned} K_f &= P C^T R_{mn}^{-1} \text{ where:} \\ \dot{P} &= A P + P A^T + G Q_{pn} G^T - P C^T R_{mn}^{-1} C P = 0 \end{aligned} \quad (3.1.6, 7)$$

The mean square reconstruction error is shown in equation 3.1.8

$$J = \lim(t \rightarrow \infty) E[(\underline{x}(t) - \hat{\underline{x}}(t))^T W (\underline{x}(t) - \hat{\underline{x}}(t))] = \text{trace}[P W] \quad (3.1.8)$$

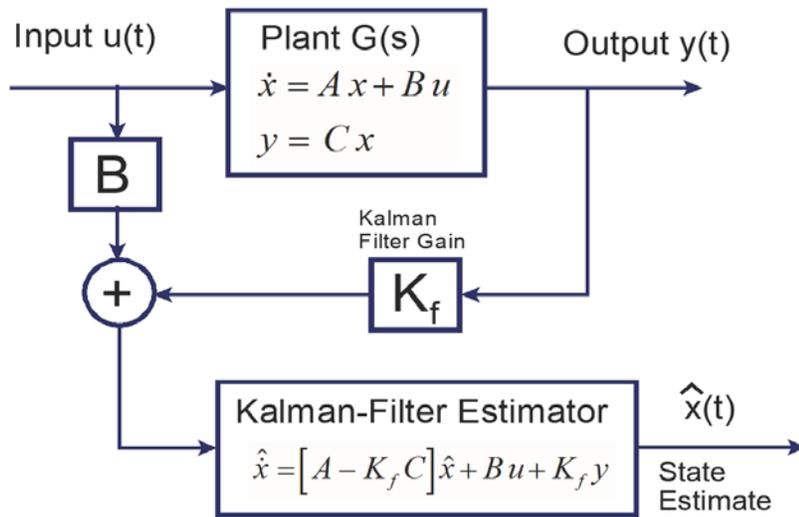


Figure 3.1 Functional Block Diagram of the Kalman-Filter Steady-State Observer

The optimal observer described provides a compromise between the speed of state reconstruction and the immunity to measurement noise. The balance between these two properties is determined by the magnitudes of the white noise intensity matrices Q_{pn} and R_{mn} that can be adjusted to satisfy design requirements. Decreasing R_{mn} and increasing Q_{pn} improves the speed of state reconstruction and shifts the observer poles further to the left side of the complex plane but the observer becomes more vulnerable to observation noise.

3.2 The Discrete-Time Kalman-Bucy Filter

The steady-state Kalman-Filter estimator for the discrete plant is obtained in a similar fashion. The dynamic model for the discrete plant is defined by the difference equations 3.2.1, where the vectors x , w , v , and y are defined as in the continuous equation 3.1.1.

$$\begin{aligned} x(k+1) &= A x(k) + B u(k) + G w(k) \\ y(k) &= C x(k) + v(k) \end{aligned} \quad (3.2.1)$$

The discrete optimal Kalman Filter estimator problem is to construct an estimate of the state $\hat{x}(k)$ from previous measurements of the output vector $[y(0), y(1) \dots y(k-1)]$ such that the quantity J in equation 3.2.2 is minimized, where E denotes the expected value, and W is $(n \times n)$ positive semi-definite matrix.

$$J = \lim(k_0 \rightarrow \infty) E \left[(\underline{x}(k) - \hat{\underline{x}}(k))^T W (\underline{x}(k) - \hat{\underline{x}}(k)) \right] \quad (3.2.2)$$

The solution exists when the pair (A^T, C^T) is stabilizable and the pair (A, D) is detectable
Where: $D D^T = G Q_{pn} G^T$

After initializing with the expected value of the initial state: $\hat{\underline{x}}(0) = E[\underline{x}(0)]$, the state estimate is computed by the following difference equation 3.2.4. This equation can also be written as in 3.2.5 to show that the inputs to the estimator are the plant control vector $\underline{u}(k)$ and the measurements $\underline{y}(k)$.

$$\begin{aligned} \hat{\underline{x}}(k+1) &= A \hat{\underline{x}}(k) + B \underline{u}(k) + K_f \left[\underline{y}(k) - C \hat{\underline{x}}(k) \right] \\ \hat{\underline{x}}(k+1) &= \left[A - K_f C \right] \hat{\underline{x}}(k) + B \underline{u}(k) + K_f \underline{y}(k) \end{aligned} \quad (3.2.4, 5)$$

The Kalman-Filter gain K_f is calculated from equation (3.2.6), where matrix P represents the steady-state variance of the state-vector reconstruction error. It is symmetric positive semi-definite and is obtained by solving asymptotically the recursive Riccati Equation 3.2.7.

$$\begin{aligned} K_f &= A P C^T \left(R_{mn} + C P C^T \right)^{-1} \text{ where:} \\ P_{k+1} &= \left(A - K_f C \right)^T P_k \left(A - K_f C \right) + G Q_{pn} G^T + K_f R K_f^T \end{aligned} \quad (3.2.6, 7)$$

The initial state of the estimator must be set equal to the plant state: $\hat{\underline{x}}(0) = \underline{x}_0$. The following result is also true

$$J = \lim(k_0 \rightarrow \infty) E \left[(\underline{x}(k) - \hat{\underline{x}}(k))^T W (\underline{x}(k) - \hat{\underline{x}}(k)) \right] = \text{trace}[P W]$$

4. Linear Quadratic Gaussian Output Feedback Control

The Linear Quadratic state-feedback controller and the Kalman-Filter results obtained from Sections 1 and 3 are now combined together to create an output feedback dynamic controller. Let us again consider the state-space plant model that we want to control.

$$\begin{aligned}\dot{\underline{x}}(t) &= A \underline{x}(t) + B \underline{u}(t) \\ \underline{y}(t) &= C \underline{x}(t)\end{aligned}\tag{4.1.1}$$

Where:

$\underline{x}(t)$ is the plant state of dimension n
 $\underline{u}(t)$ is the plant control input of dimension m
 $\underline{y}(t)$ is the plant output of dimension r

The optimal steady-state, state-feedback control $\underline{u}^o(t) = -K_c \underline{x}(t)$ was derived in Section 1, and the Kalman-Filter observer was described in Section 3.

$$\begin{aligned}\hat{\underline{x}}(t) &= A \hat{\underline{x}}(t) + B \underline{u}(t) + K_f [y(t) - C \hat{\underline{x}}(t)] \\ \dot{\hat{\underline{x}}}(t) &= [A - K_f C] \hat{\underline{x}}(t) + B \underline{u}(t) + K_f y(t)\end{aligned}\tag{4.1.3, 4}$$

Since the state vector $\underline{x}(t)$ is not directly available for measurement, we will use the estimated state vector and apply the control feedback through the estimated state rather than the actual state

$$\underline{u}^o(t) = -K_c \hat{\underline{x}}(t)\tag{4.1.5}$$

The block diagram in Figure 4.1a shows the state-estimator and the state-feedback controller operating in closed loop form around the plant. Figure 4.1.b is the same system but the observer and the state-feedback gain are combined together as a single dynamic control system that provides feedback from the plant output instead of the states. The closed loop dynamic model is obtained by combining the plant and Kalman-Filter equations in a $(2n \times 2n)$ system 4.1.6.

$$\begin{pmatrix} \dot{\underline{x}} \\ \dot{\hat{\underline{x}}} \end{pmatrix} = \begin{bmatrix} A & -B K_c \\ K_f C & A - K_f C - B K_c \end{bmatrix} \begin{pmatrix} \underline{x} \\ \hat{\underline{x}} \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} \underline{u}\tag{4.1.6}$$

After considering the state reconstruction error $\underline{e}(t) = \underline{x}(t) - \hat{\underline{x}}(t)$ we obtain equation 4.1.7

$$\begin{pmatrix} \dot{\underline{x}} \\ \dot{\underline{e}} \end{pmatrix} = \begin{bmatrix} A - B K_c & -B K_c \\ 0 & A - K_f C \end{bmatrix} \begin{pmatrix} \underline{x} \\ \underline{e} \end{pmatrix}\tag{4.1.7}$$

The eigenvalues of the system in equation 4.1.7 consist of the eigenvalues of $\det(sI - A + BK_c)$ plus the eigenvalues of $\det(sI - A + K_f C)$. Consequently the closed loop system comprises the eigenvalues of the optimal controller under state feedback, plus the eigenvalues of the observer. This is an important principle called the "separation principle", because, if we design a stable state-feedback controller and an asymptotically stable observer independently, the resulting interconnection system, equations (4.1.6) or (4.1.7), is an asymptotically stable system.

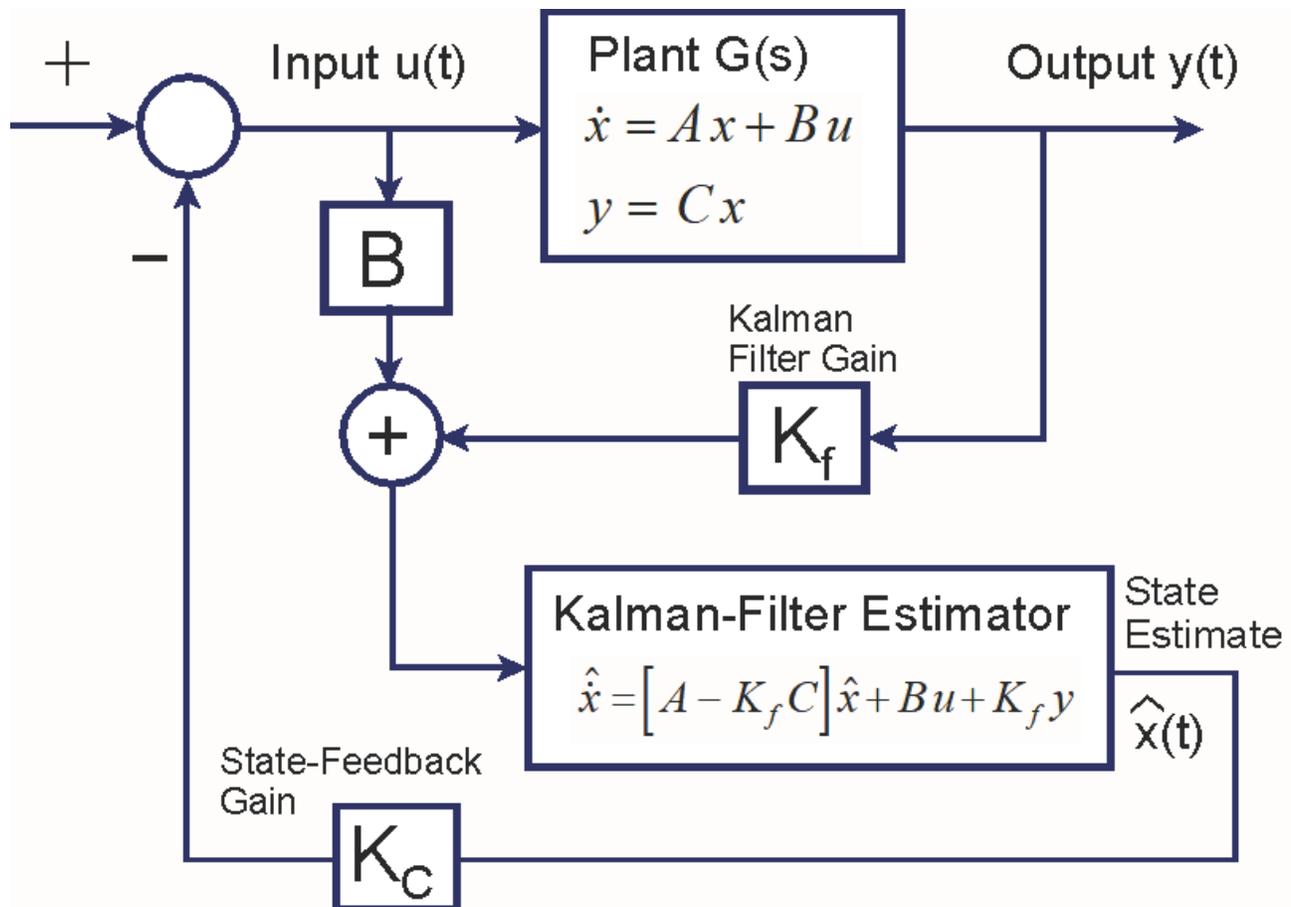


Figure 4.1a Structure of the Output Feedback Dynamic Controller Consisting of the Kalman-Filter Estimator and the State-Feedback Gain K_c

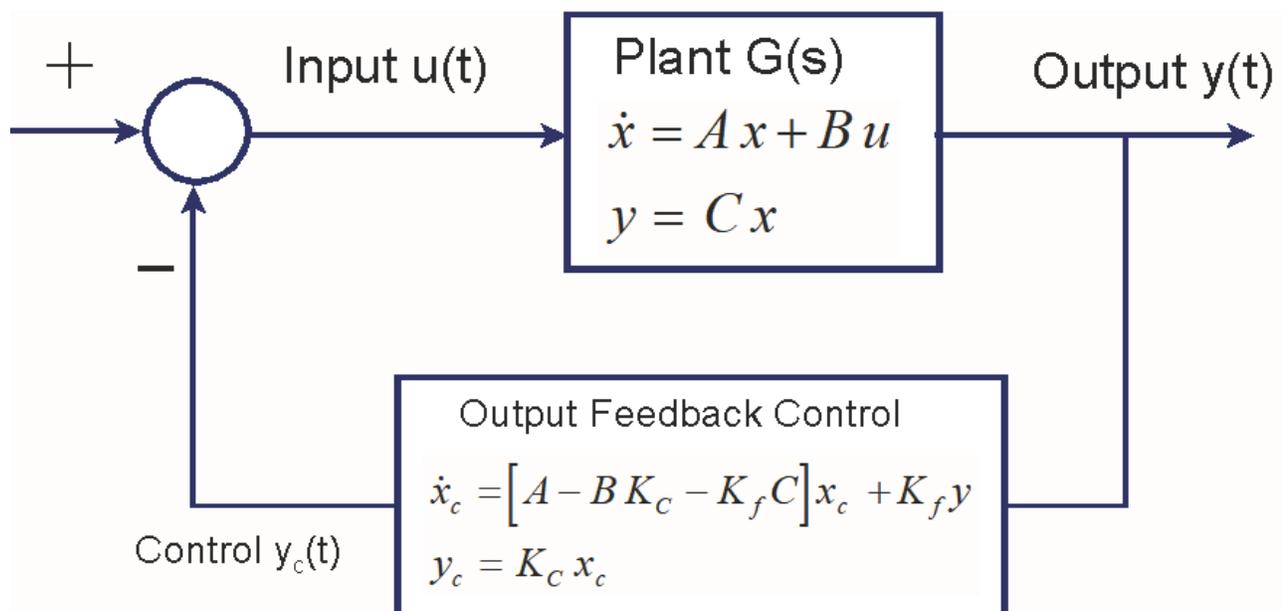


Figure 4.1b Closed-Loop System with Dynamic Controller in State-Space Form Providing Feedback from the Plant Output

5. Linear Quadratic Control Program

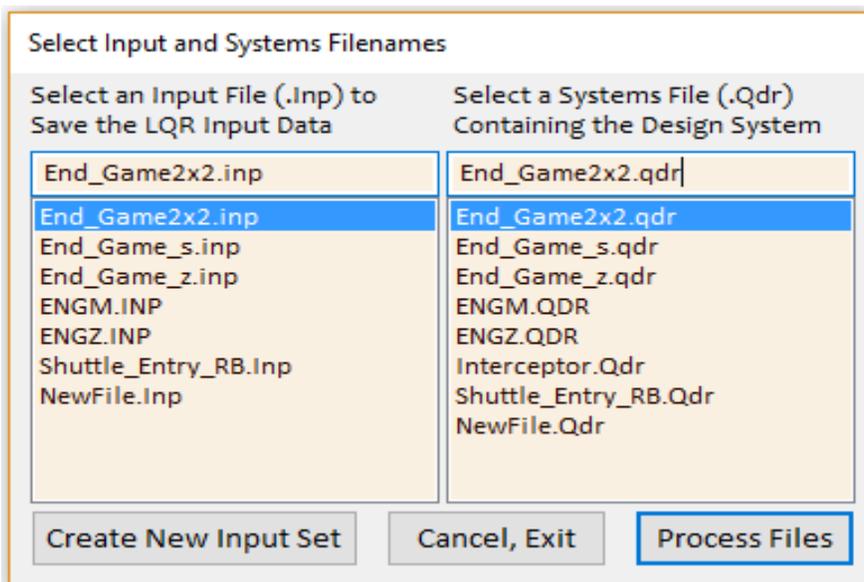
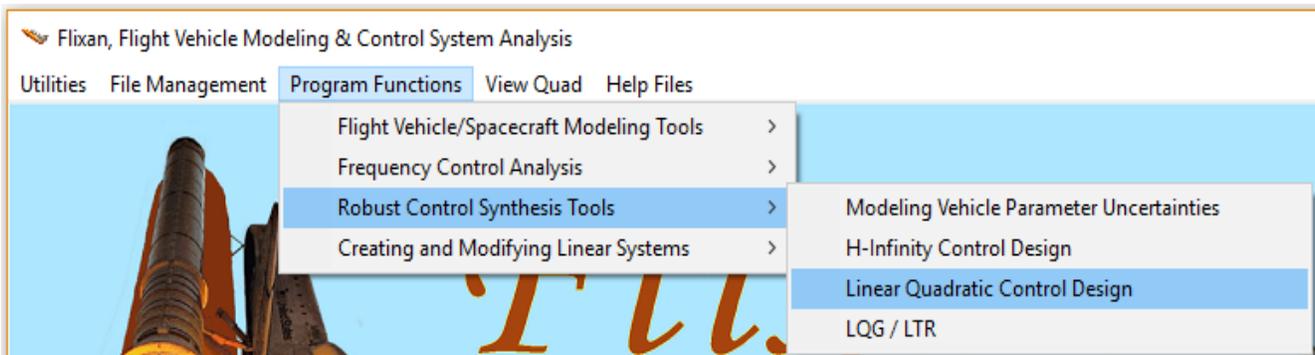
The Flixan Linear Quadratic Control program implements the four LQG functions described in Sections 1 through 4. The user must provide the plant model $G(s)$ and the weight matrices described which must be included in a systems file (.Qdr). The matrices can also be entered interactively. The program calculates the control matrix K_c , the estimator gain K_f or the LQG dynamic control system $K(s)$ and saves them in the same systems file. It runs either interactively or in batch mode. When in batch mode it processes input datasets from an already created input file (.Inp). The dataset of an operation is automatically created and saved in the input file after running it interactively the first time. It can be reprocessed multiple times in batch mode which is much faster than interactively. Typically the user runs initially the Flixan programs interactively to create the datasets and then when he needs to make a modification in the data he may reprocess the datasets in batch mode, either each set individually or the entire file using a batch set. The program consists of the following options:

1. The Asymptotic LQR design for a continuous or discrete time plant described in Section 1. The program reads the continuous $G(s)$ or discrete $G(z)$ plant state-space model from the systems file, the output weighting matrix Q_c , and the control weighting matrix R_c . It solves either the continuous or the discrete LQR problem depending on the plant sampling period, which is zero when the plant is continuous. It calculates the steady-state LQR state-feedback gain K_c and saves it in the systems file (.Qdr). The user may choose between two algorithms for solving the asymptotic Riccati equation.
2. The Transient LQR design for a continuous or discrete time plant described in Section 2. The program reads the continuous $G(s)$ or discrete $G(z)$ plant state-space model from the systems file, the $(r \times r)$ output weighting matrix Q_c , the $(m \times m)$ control weighting matrix R_c , and the $(n \times n)$ weighting matrix P_1 that penalizes the state vector at the terminal time t_f . It requires also the initial time-to-go before the final time, and the number of points to calculate the state-feedback gains (only when the plant is continuous). It calculates the time-varying gain matrix $K_c(t)$ and saves it as a function of time-to-go in Excel format in file "Gains.Txt". Tgo is in the first column and the gain matrix is printed in rows.
3. The Kalman-Filter State Estimator for a continuous or discrete time plant described in Section 3. The program reads the continuous $G(s)$ or discrete $G(z)$ plant state-space model from the systems file, the $(n \times l)$ input noise matrix G , the $(r \times r)$ process noise intensity matrix Q_{pn} , and the $(m \times m)$ measurement noise covariance matrix R_{mn} . It solves either the continuous or the discrete KF observer problem, calculates the $(n \times r)$ Kalman-Filter gain K_f and saves it in the systems file (.Qdr).
4. The Dynamic Output Feedback Controller that is described in Section 4. The program combines the results obtained in steps 1 and 3, which are: the state-feedback gain K_c and the Kalman-Filter gain K_f , to synthesize a steady-state control system in the situation when the state-vector is not available for measurement. It reads the two matrices from the systems file (.Qdr), calculates the output-feedback controller $K(s)$ in state-space form and saves it in the same systems file.

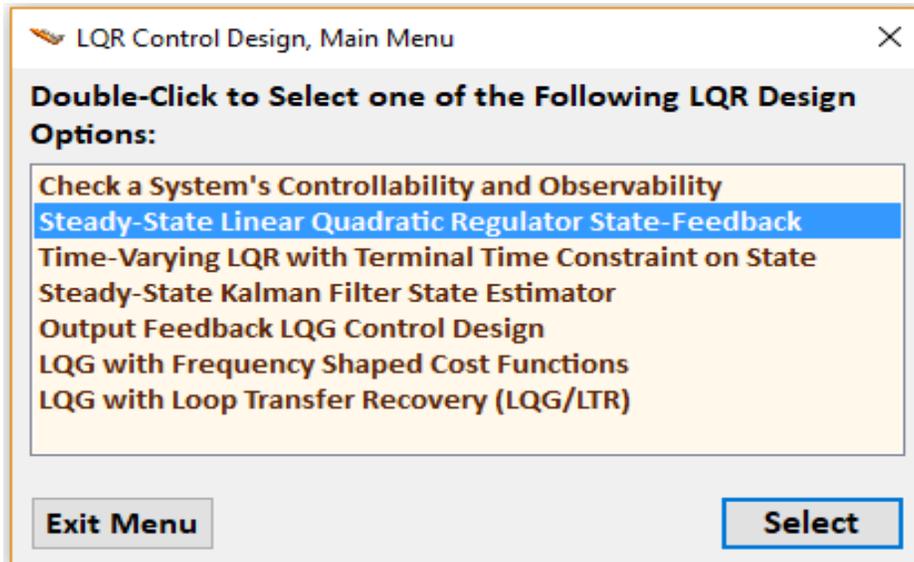
The in-between program calculations, such as matrix P , errors in the Riccati solution, closed-loop system eigenvalues, etc. are saved in file "LQC.Out" after execution. The analyst may check this file to make sure that no errors have occurred, eigenvalues are stable, matrix P is symmetric, etc. It is also important to check the plant's controllability and observability because the success of the solutions depends on that. It is the first option in the menu of the Linear Quadratic Control design program and it is only available in the interactive version when you begin analyzing the system, since it is not necessary to rerun it interactively when you reprocess the dataset in batch mode.

5.1 Running the Program Interactively

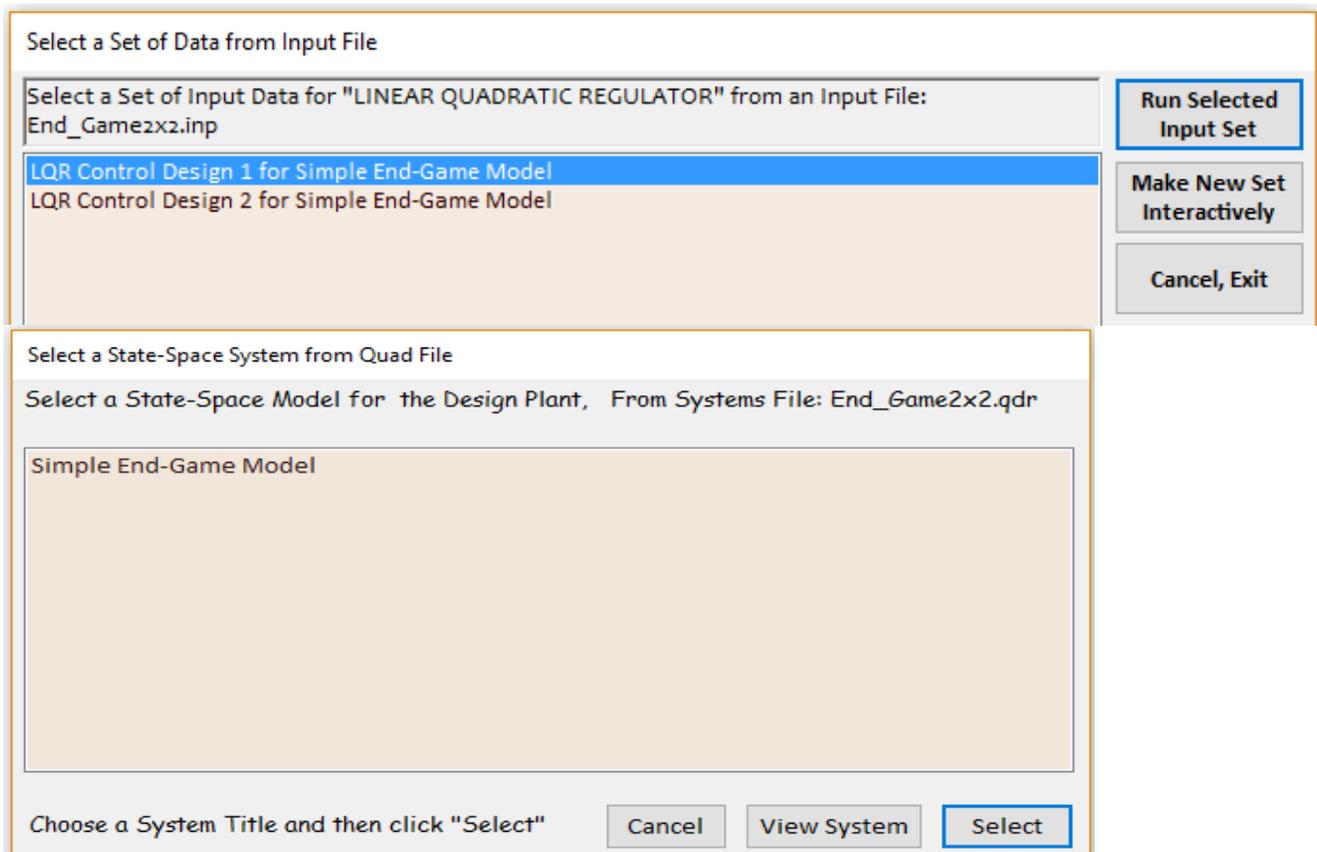
The Linear Quadratic Control design program is selected from the Flixan main menu by going to "Program Functions", "Robust Control Synthesis Tools", and then "Linear Quadratic Control Design", as shown below. Select the input filename to save the operation dataset and the systems filename where it will read and write the systems and matrices, and click on "Process Files".



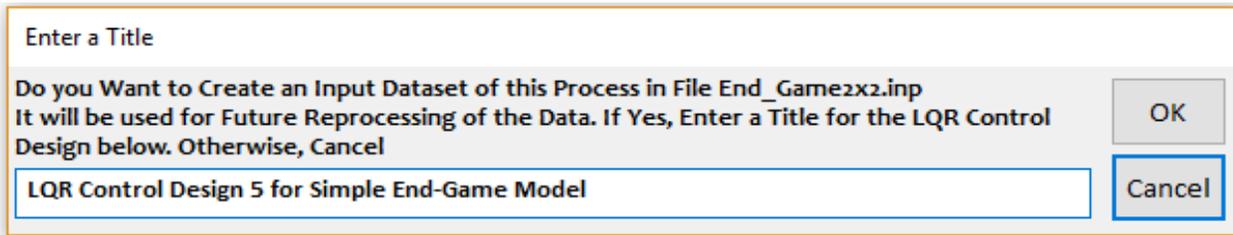
The main LQR control design menu includes several options. Select one of the options, such as: Steady-State LQR design in this case, and click on "Select".



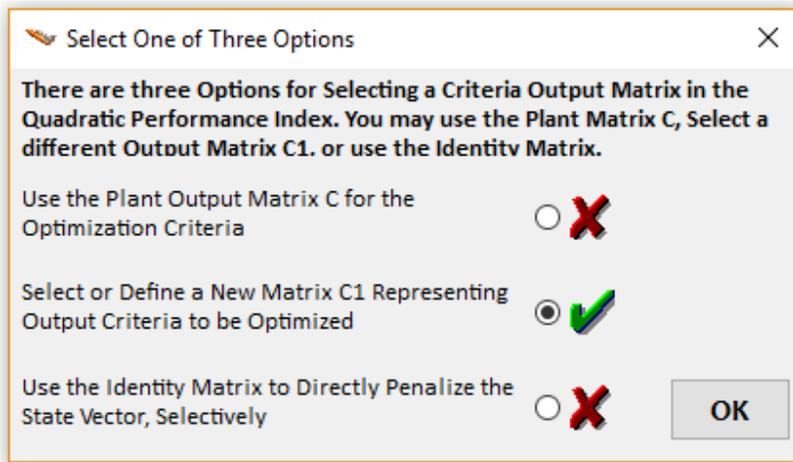
In this example the input file already contains 2 LQR design datasets. You can either process one of the 2 existing datasets or you can create a new dataset by clicking on "Make a New Set Interactively". Choose the second option and from the next menu select the title of the plant model that will be stabilized by LQR, "Simple End-Game Model" in this example.



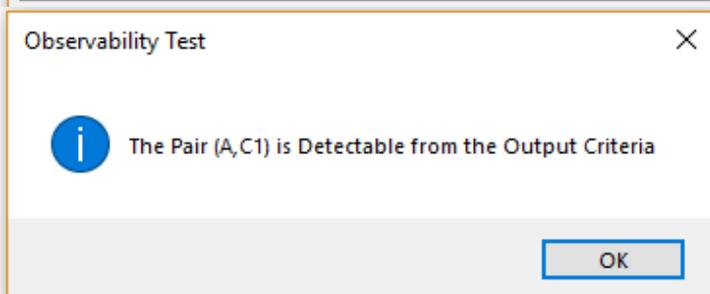
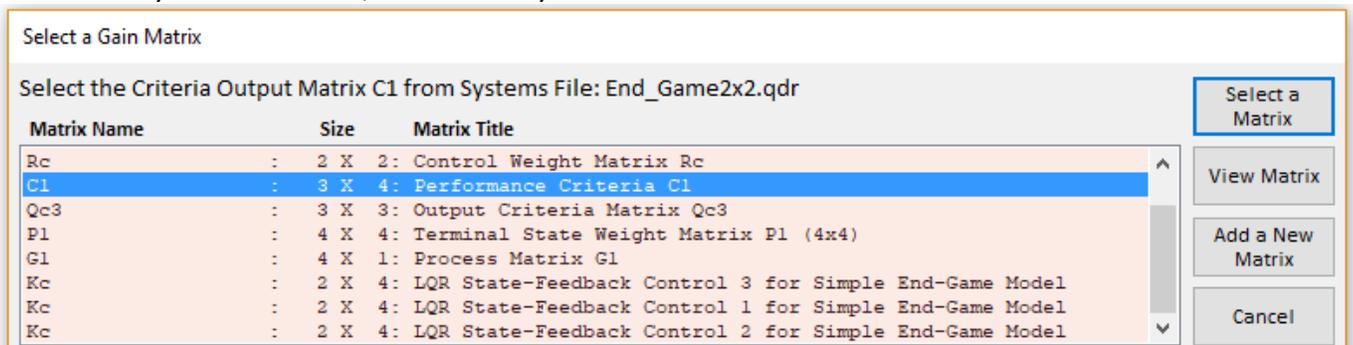
The new LQR design dataset like all datasets requires a title. Enter its title in the following dialog and click "OK". It can be used to reprocess this operation in the future when you run the program in batch mode.



The next step is to select the output criteria to be optimized. You can either use the output matrix C, the identity matrix, or define a new set of output criteria by picking a different matrix C_1 , as shown.



The following menu is used to select the output criteria matrix C_1 from the systems file. If the matrix is not in file you may create it interactively by adding a new matrix. The program checks the system's observability from matrix C_1 , which is okay in this case.



You must also select the two weight matrices Q_c and R_c from the systems file. The (3x3) matrix Q_c penalizes the 3 criteria outputs which are specified by the output matrix C_1 , and the (2x2) matrix R_c penalizes the control inputs which are 2 in this example.

Select a Gain Matrix

Select a 3 x 3 State Weight Matrix Q_c from Systems File: End_Game2x2.qdr

Matrix Name	Size	Matrix Title
Qc4	: 4 X 4	: State Weight Matrix Qc (4x4)
Qc2	: 2 X 2	: Output Weight Matrix Qc (2x2)
Rc	: 2 X 2	: Control Weight Matrix Rc
C1	: 3 X 4	: Performance Criteria C1
Qc3	: 3 X 3	: Output Criteria Matrix Qc3
P1	: 4 X 4	: Terminal State Weight Matrix P1 (4x4)
G1	: 4 X 1	: Process Matrix G1
Kc	: 2 X 4	: LQR State-Feedback Control 3 for Simple End-Game Model

Buttons: Select a Matrix, View Matrix, Add a New Matrix, Cancel

Select a Gain Matrix

Select a 2 x 2 Control Weight Matrix R_c from Systems File: End_Game2x2.qdr

Matrix Name	Size	Matrix Title
Qc4	: 4 X 4	: State Weight Matrix Qc (4x4)
Qc2	: 2 X 2	: Output Weight Matrix Qc (2x2)
Rc	: 2 X 2	: Control Weight Matrix Rc
C1	: 3 X 4	: Performance Criteria C1
Qc3	: 3 X 3	: Output Criteria Matrix Qc3
P1	: 4 X 4	: Terminal State Weight Matrix P1 (4x4)
G1	: 4 X 1	: Process Matrix G1
Kc	: 2 X 4	: LQR State-Feedback Control 3 for Simple End-Game Model

Buttons: Select a Matrix, View Matrix, Add a New Matrix, Cancel

We must finally select the algorithm to solve the asymptotic Riccati equation. The program provides 2 options. Laub's algorithm is chosen in this case. We must also enter a title for the state-feedback gain K_c that will be saved in the systems file. All systems and matrices need a title in a (.Qdr) file.

Select One of Two Options

Select a Method to Solve the Algebraic Riccati Equation
You may either choose Laubs Method or the Asymptotic Method

Use Laubs ARE Algorithm ...

Use the Asymptotic Method ...

OK

Enter a Title for the Control Gain that will be Saved in File: End_Game2x2.qdr

OK

LQR State-Feedback Control 5 for Simple End-Game Model

The following LQR design dataset was created in the input file (.Inp) that can repeat this operation in the future. The dataset includes a label on the top: "LINEAR QUADRATIC REGULATOR ..." that specifies which Flixan program will process the dataset, and a title "LQR Control Design 5 for Simple ...". The green comments were added later. It can be used to reprocess the data in batch mode.

```

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design 5 for Simple End-Game Model
! LQR Control Design for a Simple End-Game Model using
! the Criteria Optimization Outputs from Matrix C1, the
! (3x3) Output Weight Matrix Qc3, and the (2x2) Control
! weight matrix Rc. Laub's method is used to solve the ARE
!
Plant Model Used to Design the Control System from:      Simple End-Game Model
Criteria Optimization Output is Matrix C1                Performance Criteria C1
State Penalty Weight (Qc) is Matrix:  Qc3              Output Criteria Matrix Qc3
Control Penalty Weight (Rc) is Matrix: Rc              Control Weight Matrix Rc
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kc              LQR State-Feedback Control 5 for Simple End-Game
-----

```

The (2x4) state-feedback gain matrix Kc was saved in the systems file (.Qdr) under the specified title. The comments are transferred from the input dataset to the matrix in the systems file. The definitions of the matrix inputs and outputs are determined from the plant model variables.

```

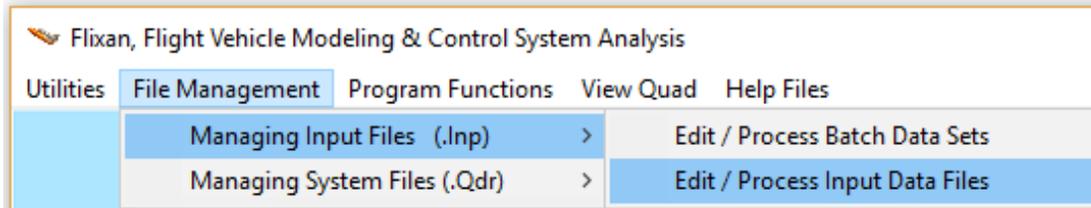
Gain Matrix for ...
LQR State-Feedback Control 5 for Simple End-Game Model
! LQR Control Design for a Simple End-Game Model using the Criteria Optimization
! Outputs from Matrix C1, the (3x3) Output Weight Matrix Qc3, and the (2x2) Control
! weight matrix Rc. Laub's method is used to solve the ARE
!
Matrix Kc          Size = 2 X 4
      1-Column      2-Column      3-Column      4-Column
1-Row -0.9999999945015E+01 -0.141873048039E+02 -0.684422068284E+00 0.906398261604E+01
2-Row 0.234488998893E-05 0.344151158201E-05 0.166220158816E-06 -0.342211034119E-08
-----
Definitions of Matrix Inputs (Columns):  4|
Relative Position
Relative Velocity
Target Acceleration
Interceptor Acceleration

Definitions of Matrix Outputs (Rows):  2
Interceptor Acceleration Command
Target Acceleration Noise
-----

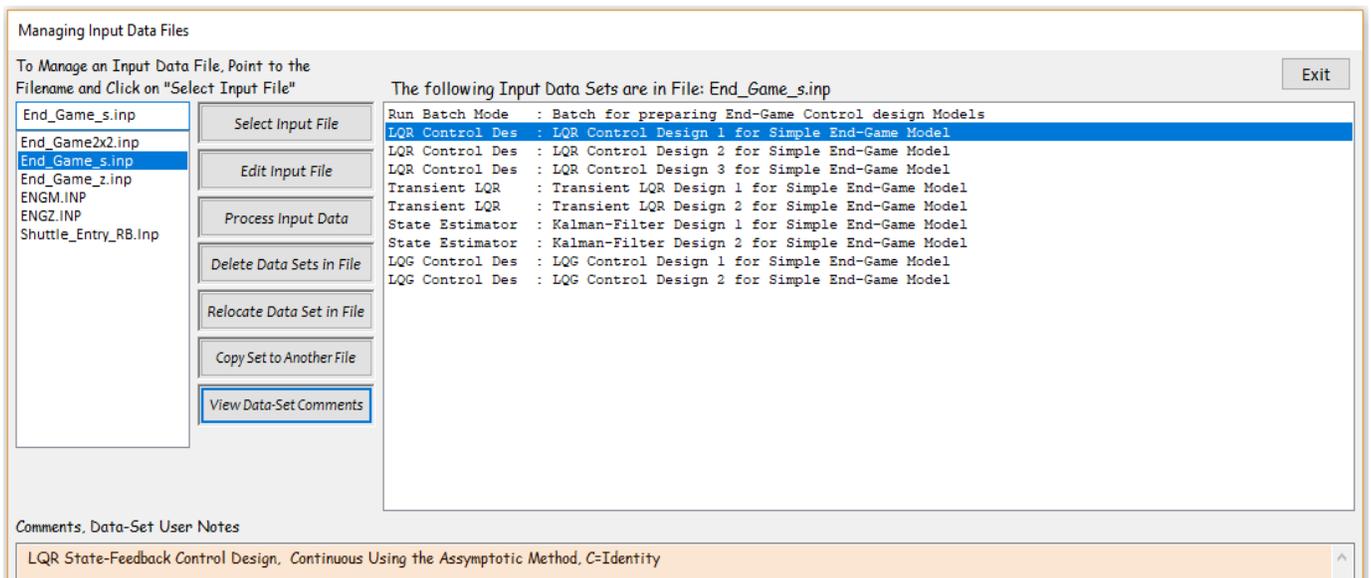
```

5.1 Running the Program in Batch Mode

To run a previously created dataset, such as an LQR design, for example, you must first select the project directory, and from the Flixan main menu, go to “File Management”, “Managing Input Files”, and then “Edit/ Process Input Data Files”, as shown below.



The input file management utility dialog comes up and from the left menu select the input file that contains the datasets for this project by clicking on “Select Input File”. The menu on the right fills with the titles of the datasets which are included in the input file. Select one of the titles, an LQR Control Design in this example, and click on “Process Input Data”. The program will calculate the LQR gain matrix K_c and save it in the systems file, as before. You may then select another dataset, such as a Transient LQR or State Estimator, and process them also. If you include a batch set, such as the one shown here at the top, you may select it to instantly process the entire input file.



6. Examples

6.1 Overhead Crane Example

In this example the plant system consists of two masses m_1 and m_2 connected with a rope. The mass m_1 is suspended from m_2 by the rope, as shown in Figure 6.1.1, representing a simple model of an overhead crane. The mass m_2 can only move along the y direction as a result of the control force F which is applied on m_2 along the y direction. Equations 6.1.1 describe the motion of the two masses

$$\begin{aligned}
 m_1 \ddot{y}_1 &= m_1 g \sin \theta \\
 m_2 \ddot{y}_2 &= F - m_1 g \sin \theta \qquad \sin \theta = \frac{x_2 - x_1}{l} \qquad (6.1.1)
 \end{aligned}$$

Where:

- g is the acceleration due to gravity
- θ is the angle of the string from vertical
- L is the length of the pendulum

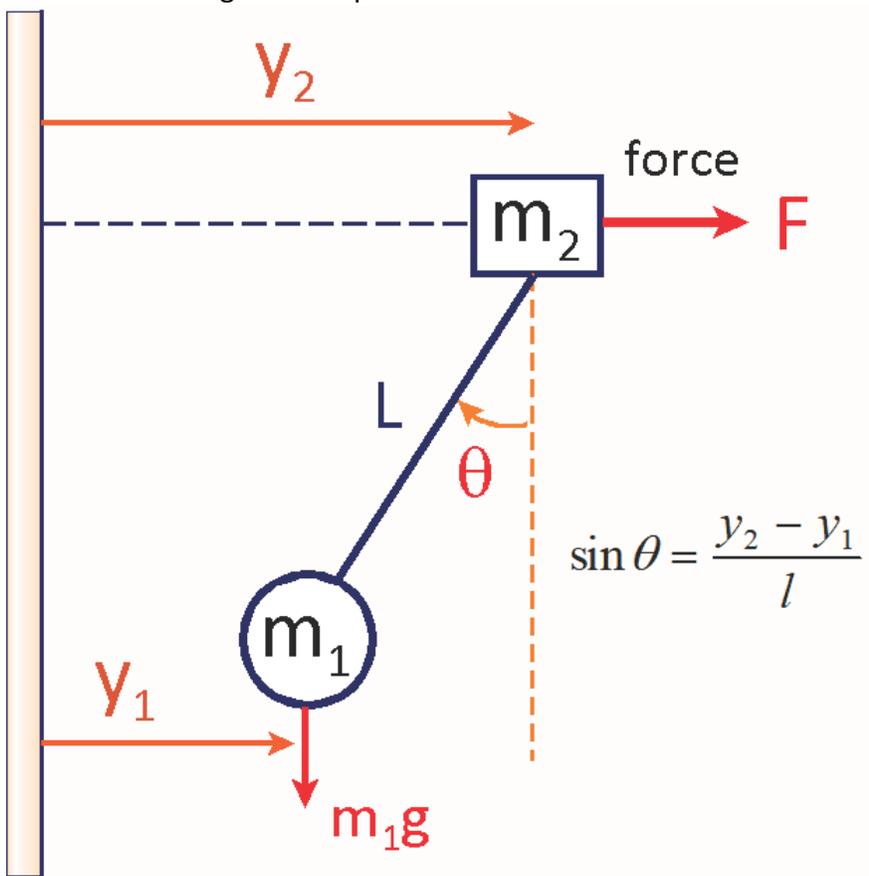


Figure 6.1.1 Simple Overhead Crane Plant Model

The design requirement for this plant is to control the position y_1 of the bottom mass m_1 by applying a control force F on the top mass m_2 and controlling its motion y_2 . From equations 6.1.1 we can write the plant equations in state space form, assuming that $g/l=1$ and $m_1=m_2$

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \underline{x} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u + G w \quad (6.1.2)$$

Where:

$\underline{x}(t)$ is the state vector, $\underline{x} = [y_1, y_2, \dot{y}_1, \dot{y}_2]$

$u(t)$ is the control force F

$w(t)$ is the process noise vector

The output vector in equation 6.1.3 consists of two deterministic measurements: the position y_1 of the mass m_1 and the pendulum angle θ of the rope from vertical.

$$\underline{z} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \end{bmatrix} \underline{x} + \underline{v} \quad (6.1.3)$$

Where: \underline{v} is a zero mean white measurement noise vector

The Analysis

In this example we shall apply the Linear Quadratic Regulator method to design a positioning control system for the bottom mass using the two output measurements (y_1, θ) to calculate the control force on m_2 . The LQR will guarantee a stable solution, but since we want to control the position y_1 of the bottom mass m_1 we will introduce one additional state in the design model (y_1 -integral) and we will design a state-feedback controller for the augmented 5-state plant. However, we cannot directly apply state-feedback because most of the states are not measurable, only y_1 and θ are. We will therefore design an estimator for the four state vector, $\underline{x} = [y_1, y_2, \dot{y}_1, \dot{y}_2]$ and apply the state-feedback from the estimated states plus the y_1 -integral which is known and it does not have to be estimated. We will use FliXan to generate the dynamic models for design and analysis, and design the LQR state-feedback and the Kalman-Filter observer. Then we will analyze the control system performance in Matlab using simulations and perform frequency response analysis.

The Flixan Files

The files for the Overhead Crane example are in subdirectory: "*Flixan\LQG\Examples\Crane*". The input file is "*Crane.Inp*" and contains the Flixan datasets for generating the plant models, calculating the steady-state LQR gains, and the Kalman-Filter. The crane design model is "*Overhead Crane Design Model*". It is augmented by including the y_1 -integral state which is intended to improve control of the y_1 position. The augmented system title is "*Crane Design Model with Y1 Integral*" and it is used to design the LQR 5-state-feedback gain $Kc1$. The 3 outputs of matrix C (y_1 -integral, y_1 , and θ) are penalized in the LQR optimization via matrix $Qc2$. The control force is penalized via the scalar Rc to achieve a satisfactory trade-off between speed of convergence and force usage. The matrices and systems are in file "*Crane.Qdr*". The batch set is used to process the entire file.

```
BATCH MODE INSTRUCTIONS .....
Batch to prepare models for the Overhead Crane Analysis and Design
! This batch Generates Dynamic Models, LQR State-Feedback Control,
! and Kalman-Filter Gain and Estimator for the Overhead Crane
!
! Retain the Old System and Matrices
Retain System      : Overhead Crane Design Model
Retain System      : Overhead Crane Analysis Model
Retain Matrix      : Output Weight Matrix Qc2
Retain Matrix      : Control Weight Matrix Rc
Retain Matrix      : State Weight Matrix Qc4
Retain Matrix      : Process Noise Covariance Matrix Qpn4
Retain Matrix      : Measurement Noise Covariance Rmn2
!
!                               Control and Estimator Design
Transf-Function    : Integrator
System Connection: Crane Design Model with Y1 Integral
LQR Control Des   : LQR Control Design 1 for Crane Design Model with Y1 Integral
State Estimator   : Kalman-Filter Design 1 for Overhead Crane Design Model
!
!                               Convert the Design and Analysis Models and Gains for Matlab Analysis
To Matlab Format   : Crane Design Model with Y1 Integral
To Matlab Format   : Overhead Crane Analysis Model
To Matlab Format   : LQR State-Feedback Control 1 for Crane Design Model with Y1 Integral
To Matlab Format   : Kalman-Filter Estimator 1 for Overhead Crane Design Model
-----

SYSTEM OF TRANSFER FUNCTIONS ...
Integrator
! Integrates the Mass-1 Displacem Y1
!
Continuous
TF. Block # 1      (1/s)                               Order of Numer, Denom= 0 1
Numer 0.0         1.0
Denom 1.0         0.0
.....
Block #, from Input #, Gain
 1      1      1.00000
.....
Outpt #, from Block #, Gain
 1      1      1.00000
.....
Definitions of Inputs = 1
Mass-1 Displacem (y1)

Definitions of Outputs = 1
Integral od Mass-1 Displacem (y1-integr)
-----
```

INTERCONNECTION OF SYSTEMS

Crane Design Model with Y1 Integral

**! Creates an Augmented plant for control Design by including the integral
! of mass-1 displacement in the states and output.**

```
!
Titles of Systems to be Combined
Title 1 Overhead Crane Design Model
Title 2 Integrator
SYSTEM INPUTS TO SUBSYSTEM 1                                Plant(s)
System Input 1 to Subsystem 1, Input 1, Gain= 1.0          Control Force
.....
SYSTEM OUTPUTS FROM SUBSYSTEM 2                                Integrator
System Output 1 from Subsystem 2, Output 1, Gain= 1.0      y1 integral
.....
SYSTEM OUTPUTS FROM SUBSYSTEM 1                                Plant Outputs
System Output 2 from Subsystem 1, Output 1, Gain= 1.0      y1 displ
System Output 3 from Subsystem 1, Output 2, Gain= 1.0      theta
.....

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 2                        Plant Outp to
Control Input                                                y1 displacem
Subsystem 1, Output 1 to Subsystem 2, Input 1, Gain= 1.0
.....
Definitions of Inputs = 1
Disturbance Force (Fdist)

Definitions of Outputs = 3
Mass-1 Displacem-Integral (y1-int)
Mass-1 Displacement (y1)
Pendulum Angle (theta)
-----
```

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN

LQR Control Design 1 for Crane Design Model with Y1 Integral

**! State-Feedback Control Design for the Augmented 5-state Crane Model
! using the output matrix C in the optimization criteria**

```
!
Plant Model Used to Design the Control System from:          Crane Design Model with Y1 Integral
Criteria Optimization Output is Matrix C
State Penalty Weight (Qc) is Matrix: Qc2                    Output Weight Matrix Qc2
Control Penalty Weight (Rc) is Matrix: Rc                    Control Weight Matrix Rc
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kc1                    LQR State-Feedback Control 1 for Crane
-----
```

KALMAN-BUCY FILTER STATE ESTIMATOR DESIGN

Kalman-Filter Design 1 for Overhead Crane Design Model

**! State Observer for the Original 4-state Crane Model, Estimating
! Positions and Velocities of the two masses from the plant output**

```
!
Plant Model Used to Design the Kalman-Filter from:          Overhead Crane Design Model
Input Process Noise Matrix (G) is the Identity
Process Noise Covariance Qpn is Matrix Qpn4                 Process Noise Covariance Matrix Qpn4
Measurement Noise Covariance is Matrix Rmn2                 Measurement Noise Covariance Rmn2
Kalman-Filter Estimator is Gain Matrix Kf1                  Kalman-Filter Estimator 1 for Overhead
-----
```

The estimator uses the original 4-state plant model: *“Overhead Crane Design Model”* which does not include the y_1 -integral. The Flixan program calculates the Kalman-Filter gain $Kf1$ which is exported to Matlab and used in the observer simulation to estimate the 4 states from the outputs y_1 and θ . The noise covariance matrices $Qpn4$ and $Rmn2$ are located in the systems file *“Crane.Qdr”*. Matlab conversion datasets are included at the bottom of the input file to create m-files for the gains and systems that can be loaded into Matlab.

```

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Overhead Crane Analysis Model
System
Analysis_Plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Crane Design Model with Y1 Integral
System
Design_Plant_Int
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Control 1 for Crane Design Model with Y1 Integral
Matrix Kc1
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Kalman-Filter Estimator 1 for Overhead Crane Design Model
Matrix Kf1
-----

```

Simulation Models

Figure 6.1.2 is a simulation model “Crane_Sim-1.mdl” used to test the state-feedback gain Kc1 directly from the four states: $\underline{x} = [y_1, y_2, \dot{y}_1, \dot{y}_2]$ which they are not measurable, but it is intended to check out the control design. Figure 6.1.3 shows the system’s response to y_1 displacement command, which is logically what a person would do naturally using common sense. First, move the top mass as fast as possible half way towards the intended position and stop for a short period waiting for the bottom mass to swing. The bottom mass does not immediately feel the motion until the pendulum angle θ is big enough. When the bottom mass swings over to the opposite extreme of the pendulum angle $-\theta$, which is very close to the intended position, the top mass is immediately moved above the target position to prevent it from oscillating further. This is essentially what the LQR control system does in Figure 6.1.3 but it also takes into consideration the limited control system bandwidth. This requires knowledge of the pendulum frequency which is captured in the design plant model and subsequently in the control design to dampen out the pendulum oscillations. Notice the “hick-up” in the y_2 response as it waits for the bottom mass to swing in the opposite side of the pendulum.

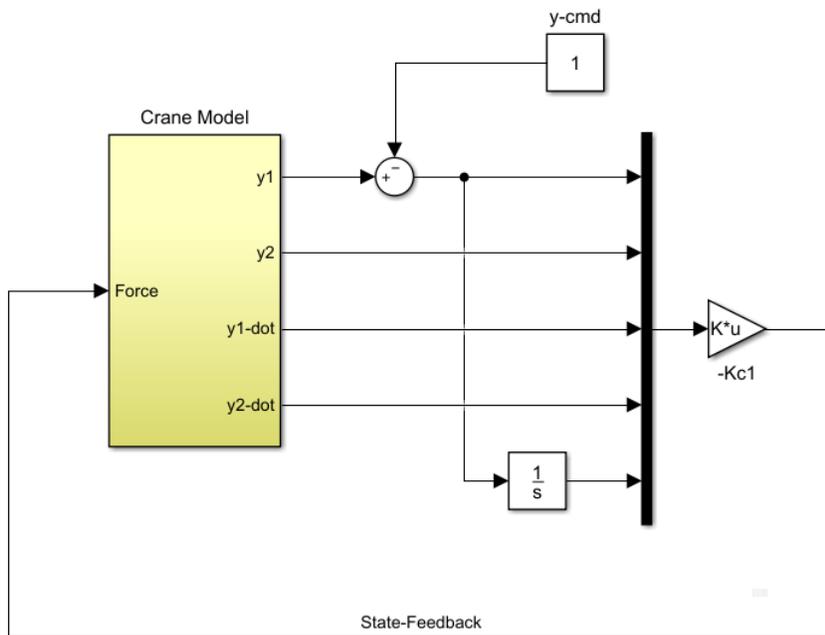


Figure 6.1.2 State-Feedback Simulation model “Crane_Sim-1.mdl”

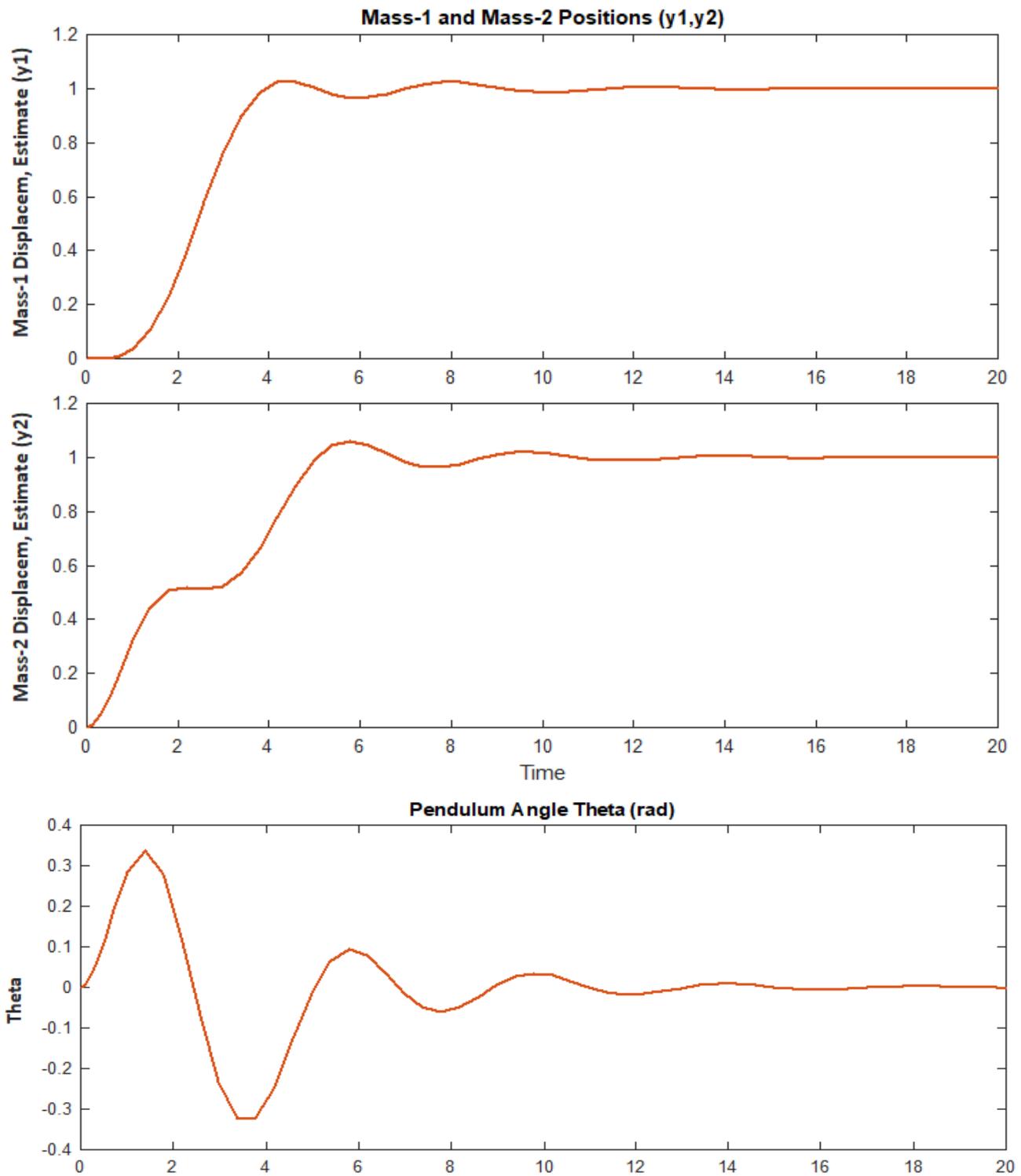


Figure 6.1.3 System's Response to a Displacement Command y_1 -command

But in reality the state vector is not available, that is why we designed the Kalman-Filter observer to estimate the states for feedback. The simulation in Figure 6.1.4 shows the control system which includes the state estimator in detail below. The inputs to the estimator are the two measurements: y_1 , and θ , and also the control force. The outputs are the four states. The y_1 -integral is not included because it is measurable. The file `init.m` loads the Flixan generated systems and matrices into Matlab for the analysis.

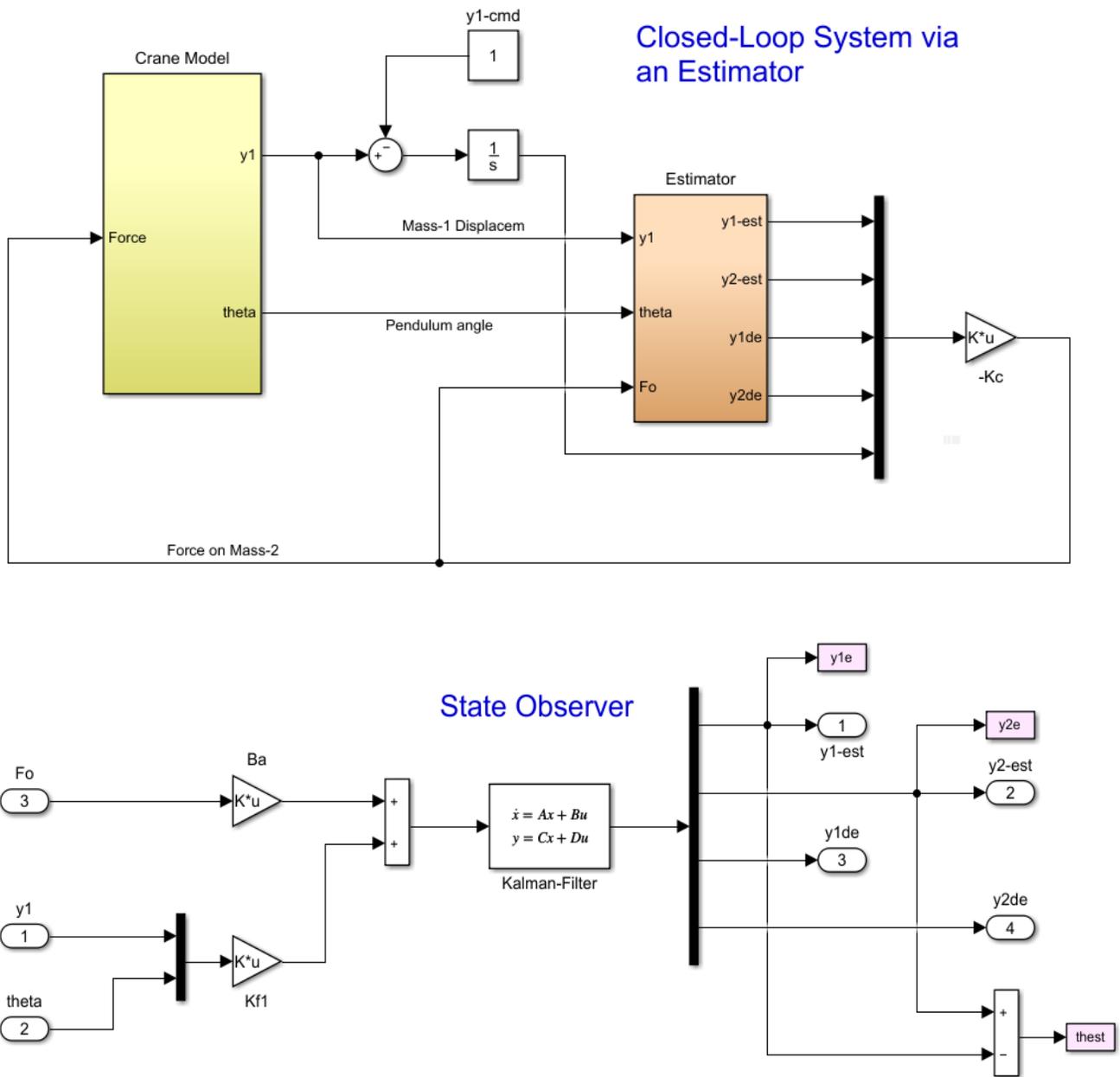


Figure 6.1.4 Output Feedback Simulation model "Crane_Sim-2.mdl" that includes the Estimator

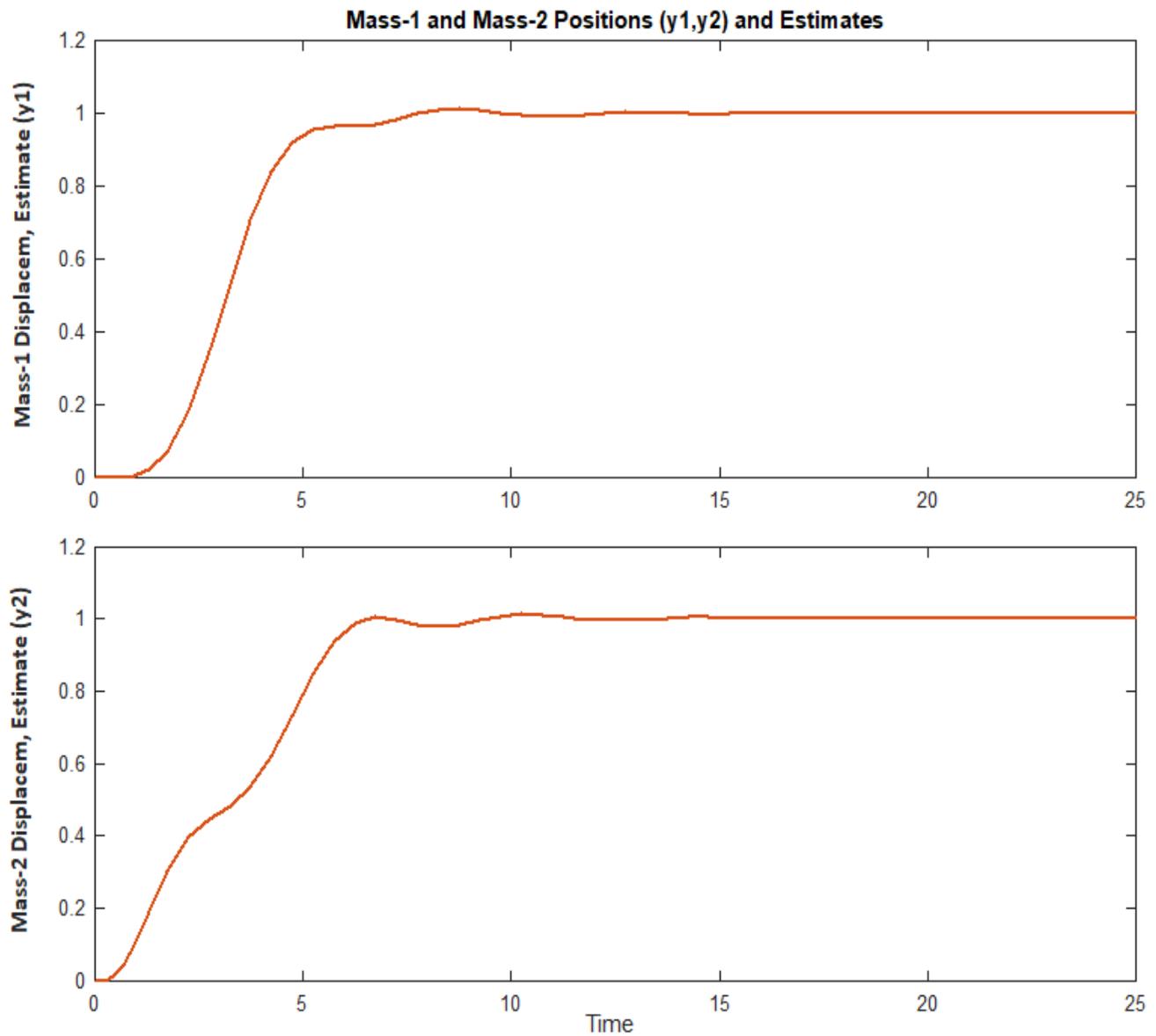


Figure 6.1.5a Response of System “Crane_Sim-2.mdl” to y_1 Displacement Command

Figure 6.1.5 shows the response of the output-feedback system which is not very different from the state-feedback system. The estimator changes slightly the response. The hick-up on the y_2 displacement is not as intense as in the state-feedback case and the oscillation damping is slightly faster. Also, the reverse force amplitude is not as high as the forward force and it is applied for a longer period.

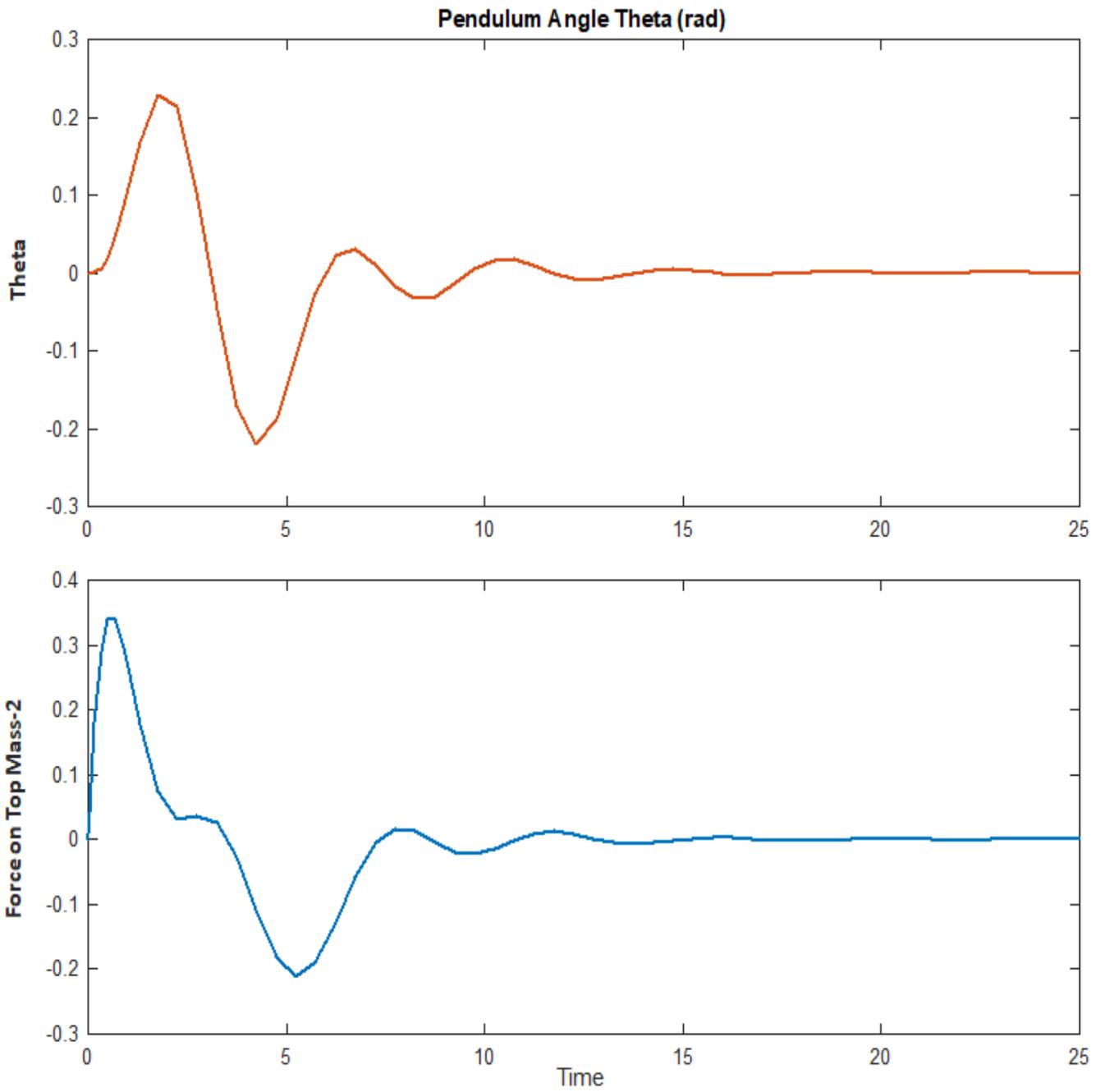


Figure 6.1.5b Response of System "Crane_Sim-2.mdl" to y1 Displacement Command

Frequency Response Analysis

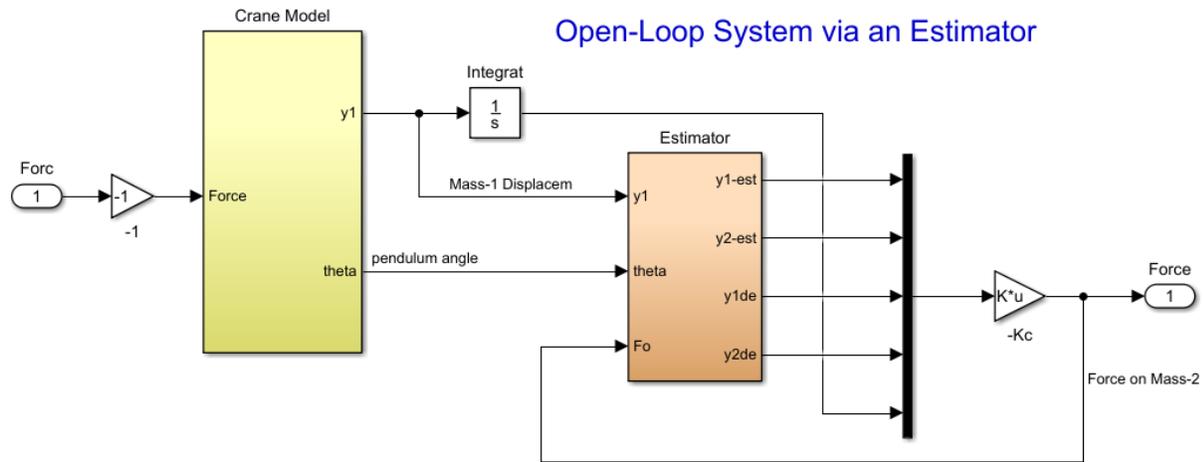
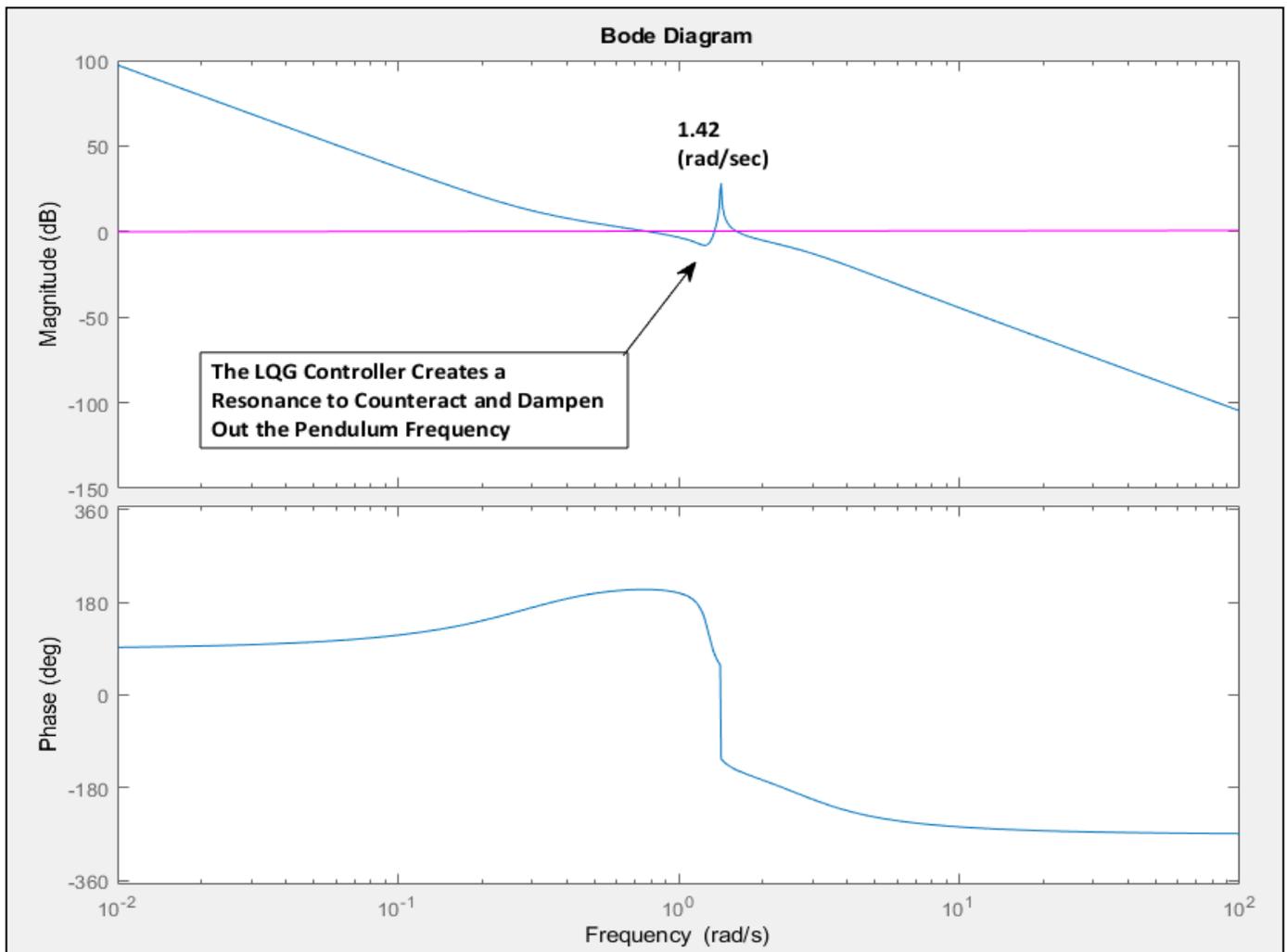


Figure 6.1.6 Frequency Response Analysis Model "Open_Loop.Mdl"



Frequency response analysis is used to check out the control system's stability in terms of gain and phase margins. The Simulink model "Open_Loop.Mdl" in Figure 6, that has the loop opened at the plant force input, is used to calculate the frequency response. Figure 6.1.7 shows the Bode and Nichols plots including the stability margins. Notice that the system has resonance of considerable amplitude at 1.42 (rad/sec) which is the pendulum frequency. This is how the control system counteracts the natural pendulum frequency by introducing an anti-resonance at the same frequency since it is designed around the plant model. Like we said earlier, the system needs to know how long to wait for the hick-up in order to counteract the natural frequency. Figure 6.1.7 below shows the phase and gain margins before and after the resonance and they are reasonable for stability.

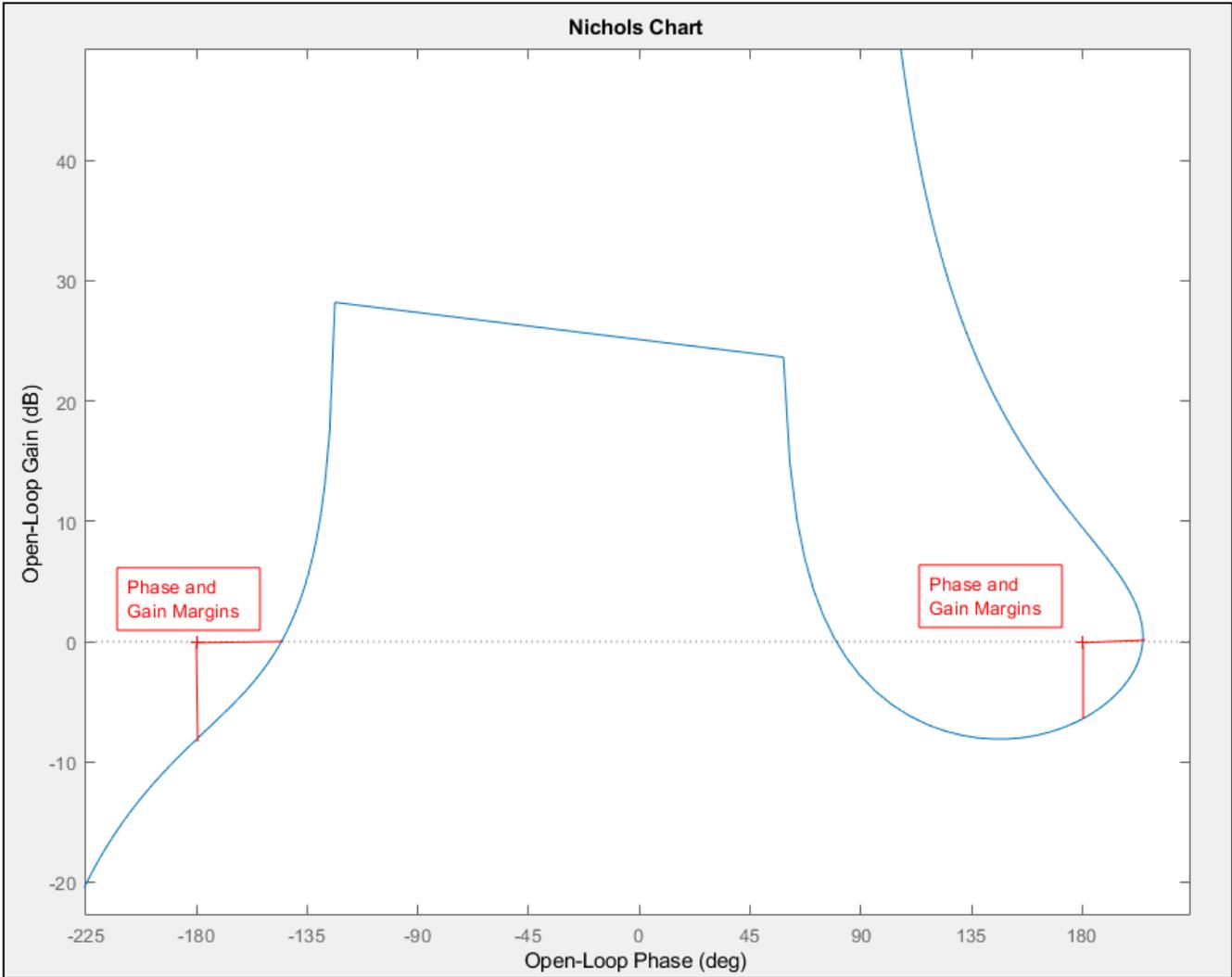
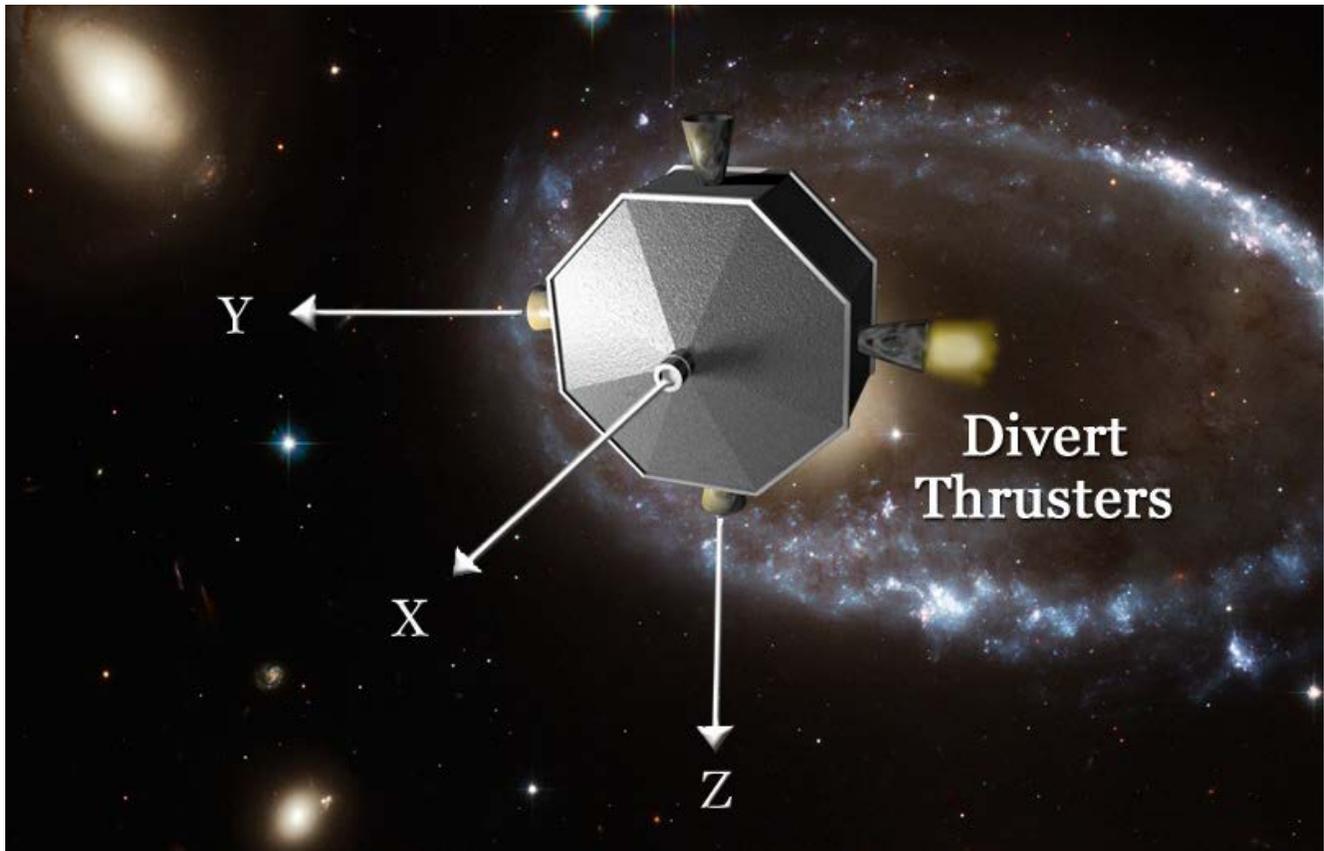


Figure 6.1.7 Open-Loop Frequency Response Analysis, Bode and Nichols Plots

6.2 Design of a Space Interceptor



In this example we will analyze a guided intercept between two space vehicles: an interceptor which is a kinetic vehicle and a target that may be a meteorite heading towards the earth, space debris, or an enemy missile. We assume that the interceptor has already been placed in a collision course with the target by a mid-course booster rocket and the vehicle is no longer accelerating but drifting towards the target. Its translational motion is controlled only in two directions (y and z) by firing divert thrusters perpendicular to the x-axis. The end-game is a dynamic engagement using closed-loop guidance to improve impact precision and probability, especially when the target is randomly accelerating in order to avoid getting hit. The interceptor uses an optical sensor to track the target and its line-of-sight (LOS) is always pointing towards the target. It has an Attitude Control System to track the target at the center of the field of view by maneuvering its attitude and aligning the x-axis with the target.

If the target is not maneuvering and if the mid-course boost was executed perfectly, the target would remain in the center of the seeker's field of view all the way to impact. If the seeker detects an error or the target is moving perpendicular to the LOS, the guidance will fire the corresponding divert thrusters to produce the necessary acceleration that will zero the error. It is assumed that the approximate relative position, velocity, and acceleration of the target are calculated from the seeker azimuth and elevation measurements, and from the target distance which is estimated from navigation and used in the End-Game algorithm.

6.2.1. End-Game Dynamic Model

The dynamic model in this Section describes the relative motion between the interceptor spacecraft and the target. The relative motion of the two spacecraft can be described by two sets of equations: one describing the relative motion along the x-direction which is along the main velocity direction and the LOS, and another set of equations that describe the motion perpendicular to the LOS. The motion along the LOS which is along the line joining the two spacecraft is uncontrollable because the interceptor has no thrust in the x-direction and the relative motion equation is only used to calculate the time to impact (t_{go}). The relative motion perpendicular to the LOS along the y and z directions is controlled by the interceptor's divert thrusters and it is identical in both perpendicular directions.

Divert Thrusters

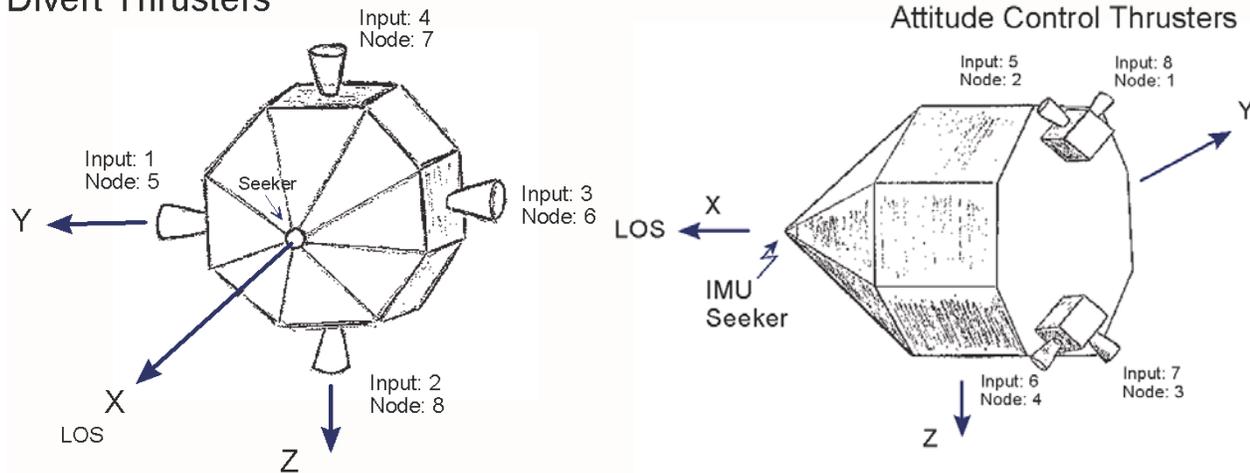


Figure 6.2.1 Interceptor Spacecraft

The time-to-go calculation t_{go} for an accelerating target in equation 6.2.1 is calculated from the estimated acceleration A_x , relative velocity V_x , and the distance to target R . If the target is not accelerating the time-to-go simplifies to: $t_{go} = R/V_x$

$$t_{go} = \frac{-V_x + \sqrt{V_x^2 - 2RA_x}}{A_x} \tag{6.2.1}$$

Equation 6.2.2 describes the relative spacecraft motion perpendicular to the LOS (either y or z directions). It is controlled by the divert thrusters that provide the interceptor acceleration A_I . It describes the motion in the local inertial frame which is defined by the position of the interceptor at the initialization time, when it initially detects the target.

$$\begin{pmatrix} \dot{S}_r \\ \dot{V}_r \\ \dot{A}_T \\ \dot{A}_I \end{pmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -W_T & 0 \\ 0 & 0 & 0 & -W_I \end{bmatrix} \begin{pmatrix} S_r \\ V_r \\ A_T \\ A_I \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & W_T \\ W_I & 0 \end{bmatrix} \begin{pmatrix} A_{Icom} \\ A_{Tcom} \end{pmatrix} \quad (6.2.2)$$

The state vector consists of four states:

- S_r is the relative vehicle position (target – interceptor)
- V_r is the relative vehicle velocity (target – interceptor)
- A_T is the target acceleration normal to the LOS
- A_I is the interceptor acceleration perpendicular to the LOS

The two inputs are:

- A_{Icom} Interceptor Acceleration Command perpendicular to the LOS
- A_{Tcom} Target Acceleration Command perpendicular to the LOS

The dynamic model in equation 6.2.2 captures the maneuverability of the two vehicles perpendicular to the LOS and introduces it in the control design. In addition to relative position and velocity it includes the target and interceptor bandwidths described by first order lags of frequencies W_T and W_I respectively, where: $W_T=5$ (rad/sec) and $W_I=100$ (rad/sec). Naturally the interceptor must have a broader bandwidth than the target because it is smaller in size. S_r and V_r are the target's position and velocity relative to the kill vehicle perpendicular to the LOS.

The control guidance of the interceptor must sense the relative motion of the target perpendicular to its x-axis using the optical sensor and apply the proper acceleration command to the thrusters in order to null-out the relative position at the estimated impact time. We can apply the Linear Quadratic Regulation to calculate the state-feedback control law that will take out the relative position at the expected impact. However, we must take into consideration two additional issues. The first issue is that there is a significant amount of noise in the measurement, especially when the distance-to-go is large. We don't want the kill vehicle to be chasing noise because it will consume its propellant fast. Therefore, we need a control system with variable bandwidth. Starting at low bandwidth for fuel efficiency and increasing it inversely proportional with time-to-go in order to improve performance near impact, where it is needed more and the signal to noise ratio is good. The second issue to be considered in the design is the uncertainty in the t_{go} calculation which is based on navigation measurements and subject to delays. We want a successful hit even if it occurs a little sooner or a little later than the expected time. One way to improve success is to reduce the relative side velocity V_r to zero a short time prior to the estimated impact time, and maintain a high gain system all the way to impact.

6.2.2. Optimal Control Design

The previously described design requirements can be captured in the performance index of the Linear Quadratic Regulator algorithm. After simplifying the state-space representation of the engagement model and assuming that all states are available for feedback, the performance index J is defined as follows

$$\begin{aligned} \dot{\underline{x}} &= A \underline{x} + B u \\ J &= \int_0^{t_f} (\underline{x}' Q \underline{x} + R u^2) dt + \underline{x}(t_f)' P_1 \underline{x}(t_f) \end{aligned} \quad (6.2.3)$$

Where: the matrices Q and P_1 and the scalar R in the performance index equation 6.2.3 are weights that trade control acceleration versus system performance to disturbances.

- Q is a positive semidefinite matrix that penalizes the state error along the trajectory
- R is a scalar that penalizes the control along the trajectory which is related to fuel
- P_1 is a positive semidefinite matrix that penalizes the state vector error only at the final time t_f

They are selected to achieve a satisfactory trade-off between fuel consumption and robustness to miss distance errors, in the presence of seeker noise and range measurement errors. During the early part of the trajectory where the solution is steady-state, we avoid penalizing much the position and velocity errors perpendicular to the x -axis in the Q matrix. The terminal position and velocity states are heavily penalized by matrix P_1 , because reducing the perpendicular components of the relative velocity to almost zero at impact, makes the optimal control law less sensitive to range errors.

The optimal state-feedback control law is obtained by synthesizing the time-varying LQR problem around the plant model of equation 6.2.2. Since we cannot control the target motion but only the interceptor's, for control design we ignore the second input to the dynamic model and keep only the A_{Icom} input. The target acceleration input will be used in the analysis. The last term in the performance index equation that includes the matrix P_1 produces the time-varying state-feedback gain. The matrix P_1 penalizes the terminal position and velocity, and by adjusting the velocity coefficient we can reduce the terminal velocity V_r to almost zero at impact, that is, in addition to the relative position S_r . The LQR solution in equation 6.2.4 calculates a time varying state-feedback gain matrix $K_c(t)$ that optimizes the performance index of equation 6.2.3 and satisfies the design requirements.

$$\begin{aligned} u^0(t) &= -R^{-1} B^T P(t) \underline{x}(t) = -K_c(t) \underline{x}(t) \\ -\dot{P}(t) &= Q - P(t) B R^{-1} B^T P(t) + P(t) A + A^T P(t) \end{aligned} \quad (6.2.4)$$

where: $0 < t < t_f$

The time-varying matrix $P(t)$ is always positive definite and symmetric and is obtained by solving the transient Riccati equation 6.2.4b. Its terminal value at impact is equal to the value of the terminal state weight matrix P_1 , i.e. $P(t_f)=P_1$. This property is used for solving the Riccati equation numerically after initializing it at the terminal time t_f and integrating backwards in time to $t=0$. The resulting control law is a time varying state-feedback that provides normal acceleration to the interceptor as a function of the four states, which at this point we assume that they are all available for feedback. In our next step we will design a Kalman-Filter to estimate the states from the system output. The same control law is used for both: the y and z axes, since the spacecraft is symmetric and there is no coupling between the y and z directions. It is important to mention that the final time t_f is not necessarily the impact time but it may be a time before impact that we wish to switch control laws, to Proportional Navigation, for example. The terminal goal in this case may be to achieve favorable conditions for PN initialization. The optimal control design boils down to choosing a satisfactory trade-off between two scalars in equation 6.2.5: the fuel weight R and the terminal state weight p . A large R penalizes the fuel usage. The larger p gets, the more the final normal relative position and velocity are reduced close zero at impact. This, however, is achieved at the expense of propellant consumption or that the control demand may exceed the maximum acceleration capability of the interceptor's thrusters. T_{gi} is a short period before impact when you expect the position and velocity to converge to zero.

$$Q = \text{diag}(0.1 \quad 0.1 \quad 0 \quad 0.1); \quad R = 1$$

$$P_1 = p \begin{bmatrix} 1 & T_{gi}/2 & 0 & 0 \\ T_{gi}/2 & T_{gi}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (6.2.5)$$

3. State Estimator Design

The LQR control law requires feedback from the state variables. However, most of the system states are not measurable and our next step is to design a state observer from the two outputs of the dynamic model, the relative position S_r and the interceptor acceleration A_I perpendicular to the LOS. In this section we will present the steady-state Kalman-Bucy filter, an observer that will be used to approximately reconstruct the state vector from the measurements so that we can apply our optimal state-feedback control law. The state observer also requires knowledge of the dynamic model in equation 6.2.2. We shall assume that the system is corrupted by two types of noise: state excitation noise, and measurement noise, as shown in equation 6.2.6. They are white noise, zero mean, and uncorrelated.

$$\begin{aligned} \dot{\underline{x}}(t) &= A \underline{x}(t) + B u(t) + G w(t) \\ \underline{y}(t) &= C \underline{x} + \underline{v}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} S_r \\ V_r \\ A_T \\ A_I \end{pmatrix} + \underline{v}(t) \end{aligned} \quad (6.2.6)$$

Where:

- $\underline{x}(t)$ is the state vector of dimension n
- $\underline{u}(t)$ is the control input vector of dimension m
- $\underline{y}(t)$ is the measurement vector of dimension r ($r \leq n$)
- $\underline{w}(t)$ is the process noise of dimension l , where ($l \leq n$) and has a covariance matrix Q_{pn} , where: $Q_{pn} = Q_{pn}' \geq 0$
- $\underline{v}(t)$ is the measurement noise with intensity $R_{mn} = R_{mn}' \geq 0$

The solution to the Kalman Filter is obtained by minimizing the quantity in equation 6.2.7, where the matrix W is $(n \times n)$ positive semi-definite. The state vector estimate $\hat{\underline{x}}$ from the KF output will converge to the actual system state \underline{x} . Figure 6.2.2 is a functional block diagram representation of the Kalman-Filter showing the output and its interconnection with the plant input \underline{u} and output \underline{y} .

$$J = \lim_{t \rightarrow \infty} E[(\underline{x}(t) - \hat{\underline{x}}(t))' W (\underline{x}(t) - \hat{\underline{x}}(t))] \quad (6.2.7)$$

The estimate is obtained by solving the following differential equation 6.2.8, where K_f is the Kalman-Filter gain. The solution exists when the pair (A',C') is stabilizable and the pair $(A, GQ_{pn}G^T)$ is detectable. The state estimate is initialized with the expected initial state vector at $t=0$, $\hat{x}(0) = E[\underline{x}(0)]$.

$$\begin{aligned} \dot{\hat{x}}(t) &= A\hat{x}(t) + Bu(t) + K_f [y(t) - C\hat{x}(t)] \\ K_f &= PC^T R_{mn}^{-1} \end{aligned} \tag{6.2.8}$$

The matrix P is symmetric positive semi-definite and it is obtained from the steady-state solution of the asymptotic Riccati equation

$$AP + PA^T + GQ_{pn}G^T - PC^T R_{mn}^{-1} CP = 0 \tag{6.2.9}$$

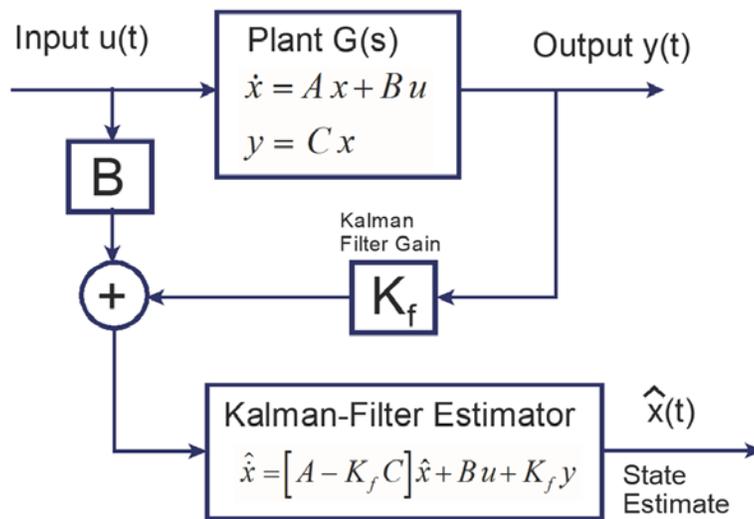


Figure 6.2.2 Kalman-Filter State-Vector Estimator

6.2.4. Continuous System Analysis

The analysis files for the continuous system in this example are located in folder: *"LQG\Examples\End-Game\Continuous"*. There is also a discrete subfolder *"Discrete"* for the discrete-time analysis using dynamic models which are discretized at 40 (msec) sampling time. The directory includes the input file *"End_Game_s.Inp"*, shown below, that contains input data for the continuous LQR design using Flixan. That is, for the steady-state LQR control, the time-varying state-feedback LQR, the Kalman-Filter, and for the steady-state output-feedback LQG design that combines the steady-state LQR gain K_c and the Kalman-Filter gain K_f to a dynamic output-feedback controller. The input file also includes a batch set for fast data processing and datasets for Matlab conversions. The systems filename *"End_Game_s.Qdr"* contains the End-Game dynamic model described in Equation 6.2.2, the control design matrices: Q_c , R_c , and P_1 , the Kalman-Filter design matrices Q_{mn} , R_{mn} , the control gain K_c and the Kalman-Filter gain K_f which are generated by the Flixan LQR design program.

BATCH MODE INSTRUCTIONS

Batch for preparing End-Game Control design Models

! This batch Generates LQR State-Feedback, Kalman-Filter and Output

! Feedback Dynamic Controller

Retain System : Simple End-Game Model
Retain Matrix : State Weight Matrix Qc (4x4)
Retain Matrix : Output Weight Matrix Qc (2x2)
Retain Matrix : Control Weight Matrix Rc
Retain Matrix : Terminal State Weight Matrix P1 (4x4)
Retain Matrix : Performance Criteria C1
Retain Matrix : Output Performance Weight Matrix Qc3
Retain Matrix : Measurement Noise Matrix Rmn (2x2)
Retain Matrix : Process Noise Matrix Qpn (4x4)
!
LQR Control Des : LQR Control Design for Simple End-Game Model
State Estimator : Kalman-Filter Design for Simple End-Game Model
LQG Control Des : LQG Control Design for Simple End-Game Model
Transient LQR : Transient LQR Design for Simple End-Game Model
!
To Matlab Format : Simple End-Game Model
To Matlab Format : LQG Control Design for Simple End-Game Model
To Matlab Format : LQR State-Feedback Control for Simple End-Game Model
To Matlab Format : Kalman-Filter Estimator for Simple End-Game Model

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN

LQR Control Design for Simple End-Game Model

! Design the State-Feedback Matrix Kc using the Output

! Criteria Matrix C= Identity

!

Plant Model Used to Design the Control System from: Simple End-Game Model
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix: Qc4 State Weight Matrix Qc (4x4)
Control Penalty Weight (Rc) is Matrix: Rc Control Weight Matrix Rc
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kc LQR State-Feedback Control for Simple End-

KALMAN-BUCY FILTER STATE ESTIMATOR DESIGN

Kalman-Filter Design for Simple End-Game Model

! Design the Kalman-Filter Gain Matrix Kf using the

! Process Noise Matrix G = Identity

!

Plant Model Used to Design the Kalman-Filter from: Simple End-Game Model
Input Process Noise Matrix (G) is the Identity
Process Noise Covariance Qpn is Matrix Qpn Process Noise Matrix Qpn (4x4)
Measurement Noise Covariance is Matrix Rmn Measurement Noise Matrix Rmn (2x2)
Kalman-Filter Estimator is Gain Matrix Kf Kalman-Filter Estimator for Simple End-Game

DYNAMIC OUTPUT FEEDBACK LQG CONTROL DESIGN

LQG Control Design for Simple End-Game Model

! Combine State-Feedback with KF Gain to Design a Linear Quadratic

! Gaussian Control System for the Plant: Simple End-Game Model

!

Plant Model Used to Design the Control System from: Simple End-Game Model
State-Feedback (Kc) is Gain Matrix : Kc LQR State-Feedback Control for Simple End-
Kalman-Filter Estim Kf is Gain Matrix: Kf Kalman-Filter Estimator for Simple End-Game

TRANSIENT LQR CONTROL DESIGN WITH TIME-VARYING GAINS

Transient LQR Design for Simple End-Game Model

! To Generate Time-Varying State-Feedback Gain Kc(t)

! as a Function of Time-to-Go, in File: Gains.dat

!

Plant Model Used to Design the Control System from: Simple End-Game Model
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix: Qc4 State Weight Matrix Qc (4x4)
Control Penalty Weight (Rc) is Matrix: Rc Control Weight Matrix Rc
Terminal State Penalty Weigh P1 Matrix P1 Terminal State Weight Matrix P1 (4x4)
Continuous LQR Solution, Final Time, Number of Points: 20.00 800.0

```

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Simple End-Game Model
System
end_game
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQG Control Design for Simple End-Game Model
System
Control
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Control for Simple End-Game Model
Matrix Kc
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Kalman-Filter Estimator for Simple End-Game Model
Matrix Kf
-----

```

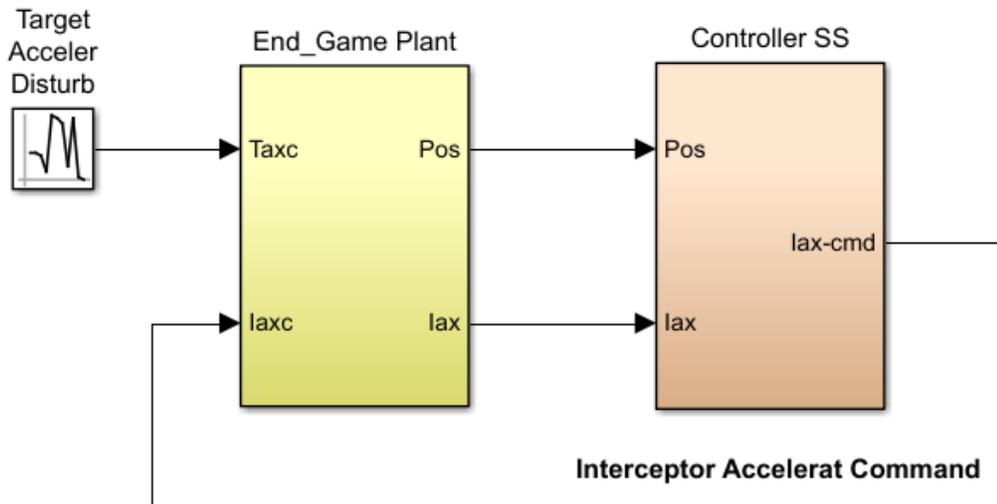
The Flixan LQR transient design program also calculates the time varying state-feedback gain $K_c(t)$ as a function of time-to-go. It requires the weight matrices: Q_c , R_c , and P_1 , the initial time-to-go (20 sec), and the number of gain calculation points (800). The gains versus time are saved in file "Gains.dat" with the time-to-go in the first column. This file is used as a look-up table in the simulation. Only the first 4 gains that correspond to the interceptor acceleration command are used in this case. The end-game dynamic model is saved in file "end_game.m", and the steady-state output-feedback control system is saved in "control.m" for Matlab analysis. The gain matrices K_c and K_f are also saved and loaded into Matlab.

6.2.4.1 Simulation Models

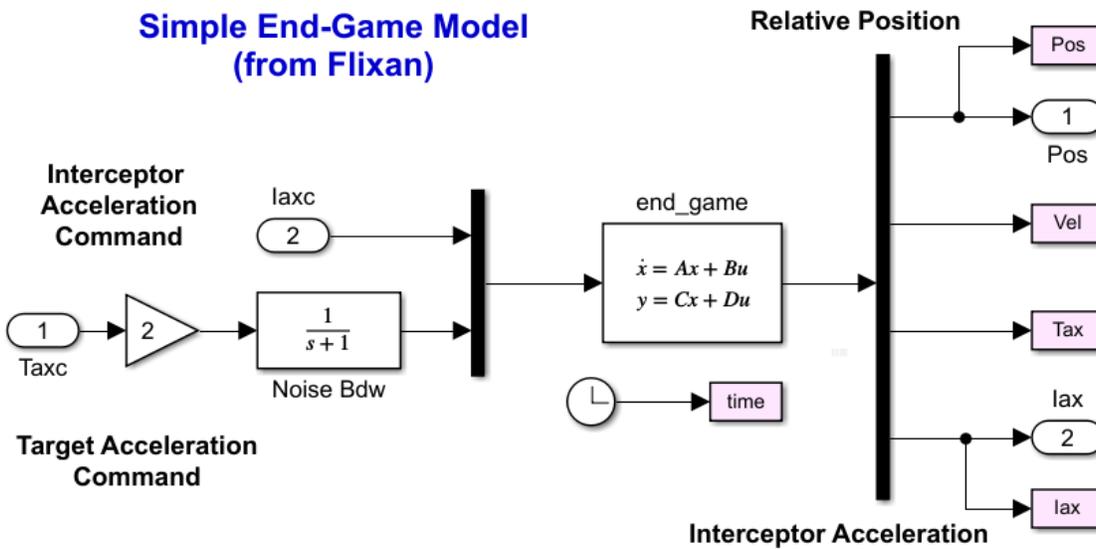
We will first analyze the steady state-performance of the spacecraft and then the transient motion with time varying state-state gains. Two simulation models were created to analyze the system's response from some initial relative condition of position, velocity and acceleration.

6.2.4.2 Continuous Steady-State Simulation

The steady-state analysis is applicable when the target is sufficiently far away from the interceptor and the control gains are constant. The Simulink model is "EndGame_Sim1s.mdl", shown in Figure 6.2.3, and it is located in the same "End-Game/ Continuous" subfolder.



Simple End-Game Model (from Flixan)



Steady-State Controller (from Flixan)

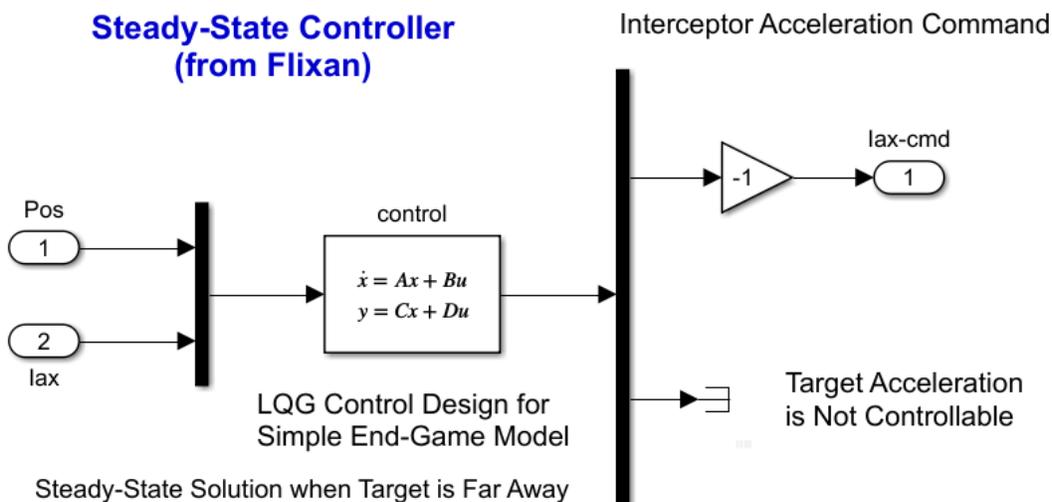


Figure 6.2.3 Steady-State Simulation Model "EndGame_Sim1s.mdl"

The plant model is the Flixan generated system “Simple End-Game Model” in file “end_game.m”. The LQG control system calculated by Flixan is: “LQG Control Design for Simple End-Game Model” in file “Control.m”. It is the dynamic system shown in Figure 6.2.4, consisting of plant model parameters (A, B, C), the steady-state feedback gain K_c and the Kalman-Filter gain K_f . The two systems and matrices are loaded into Matlab by executing the m-file “init.m”, which also initializes the state-vector \underline{x}_0 . Notice, that this configuration cannot be used in the time-varying case because $K_c(t)$ is varying.

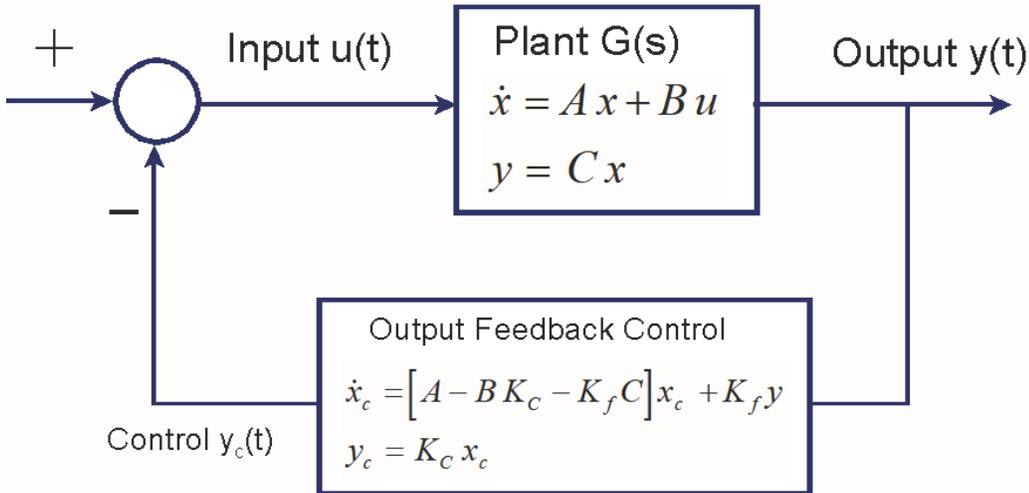


Figure 6.2.4 Steady-State LQG Output Feedback Controller/ Plant Interconnection

We can use this simulation model to calculate the system’s response to an accelerating target disturbance, as shown in Figure 6.2.5.

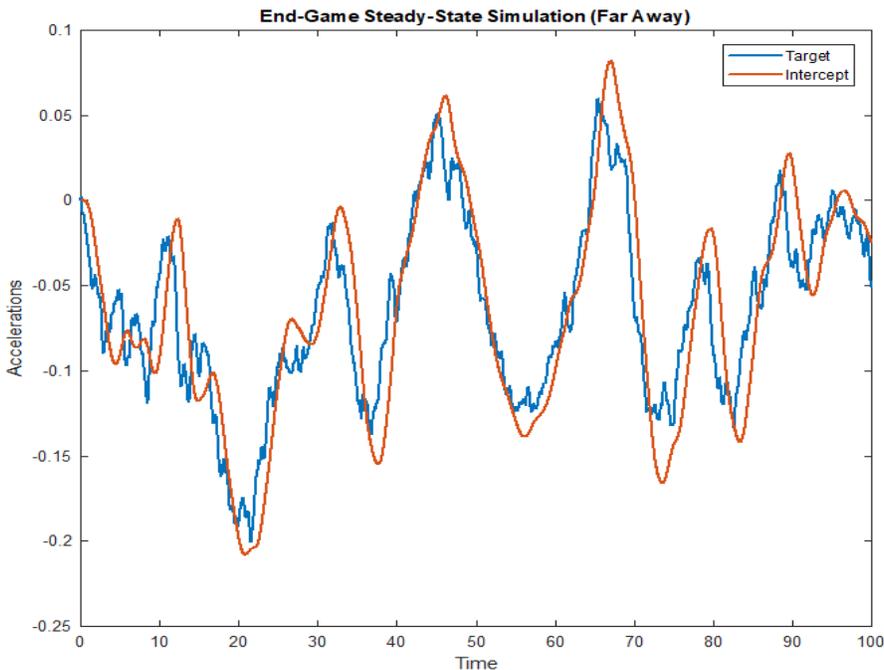


Figure 6.2.5 Interceptor Acceleration is responding to Noisy Target Accelerations

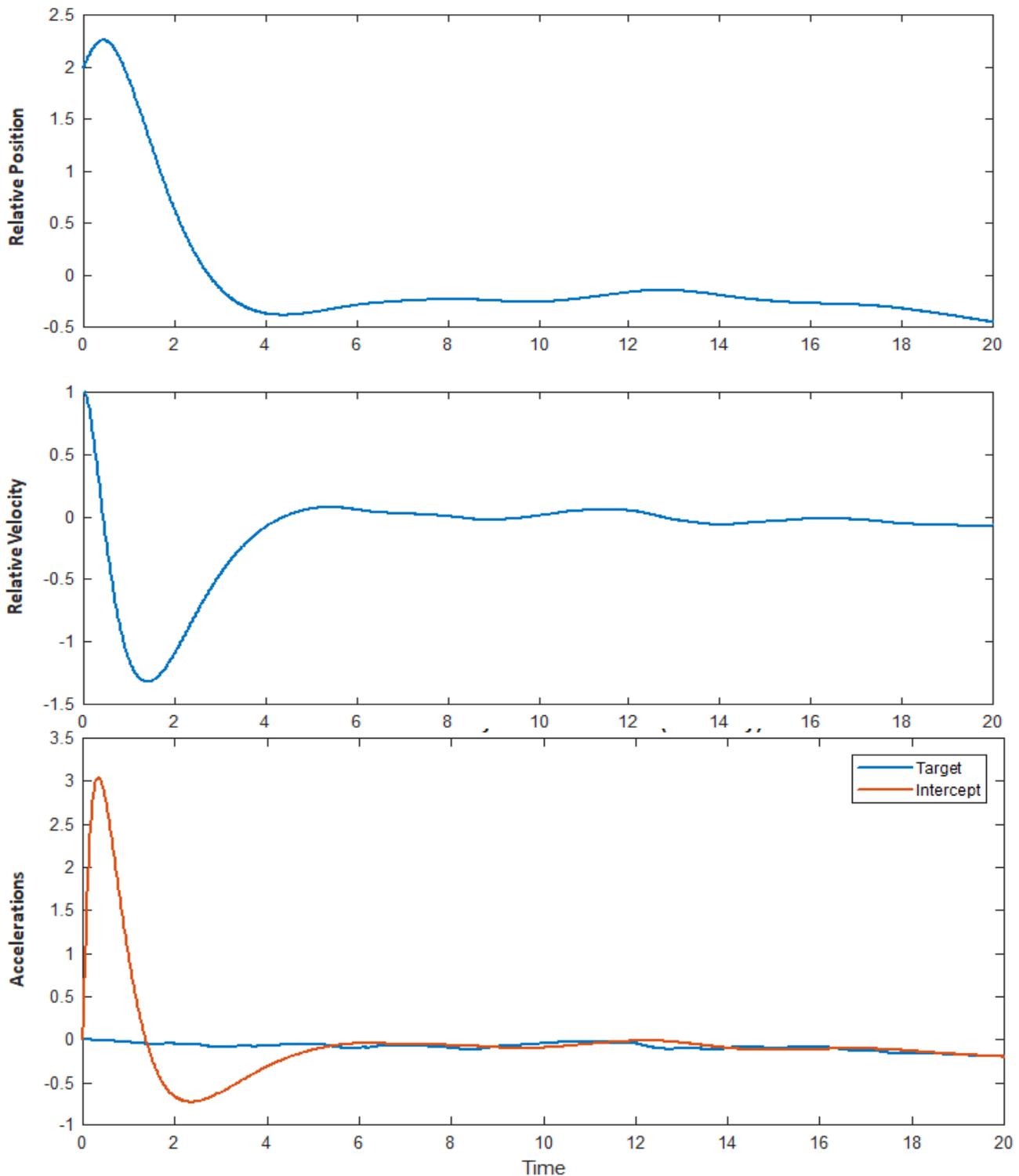
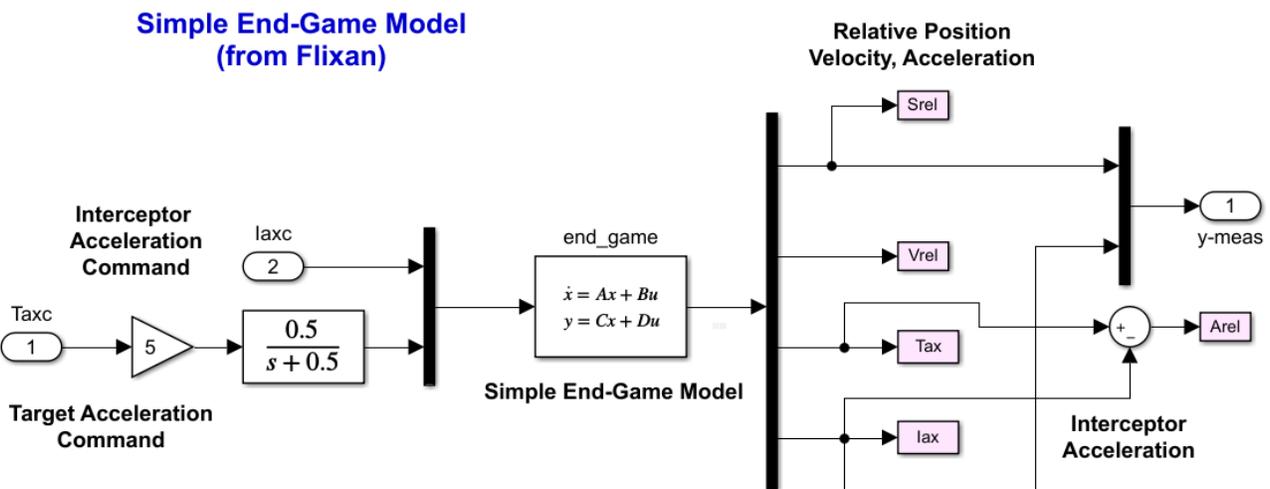
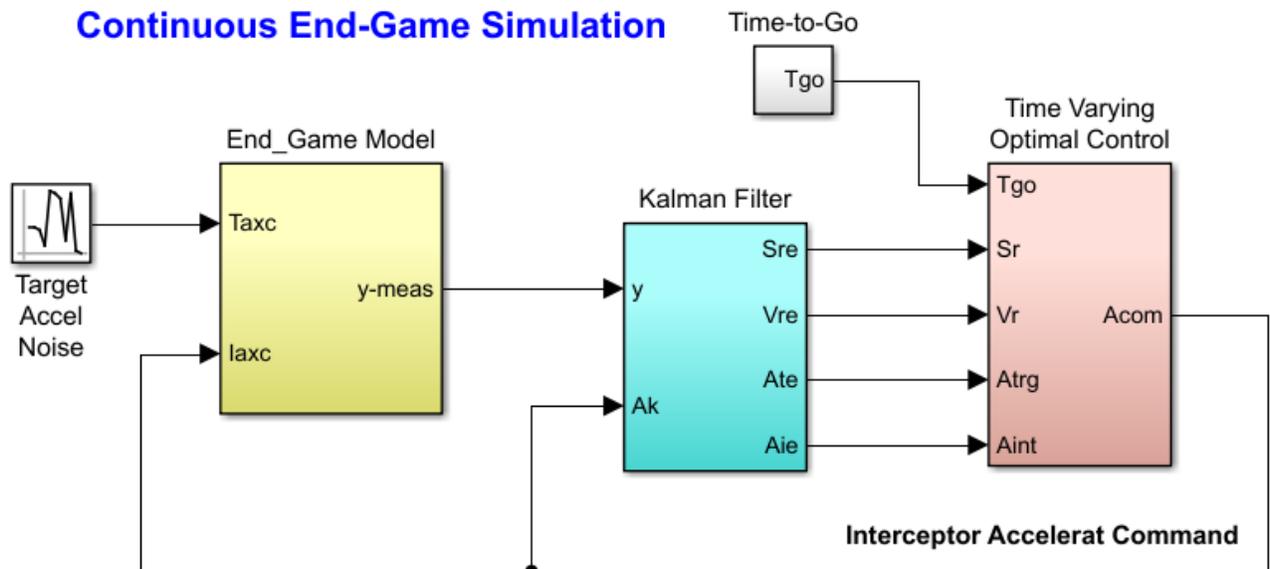


Figure 6.2.6 System's Response from Non-Zero Initial Conditions of Relative Position and Velocity

This simulation is also used to calculate the system's response to initial position and velocity errors with an accelerating target, see Figure 6.2.6. Notice that the system's response at steady-state is intentionally slow in order to save propellant since the target is still far away. This causes noticeable position error due to target acceleration.

6.2.4.3 Continuous Simulation with Time Varying Control Gains

The continuous simulation model in Figure 6.2.7 is in file "EndGame_Sim2s.mdl". It uses time-varying gains as a function of time-to-go which are loaded into Matlab look-up tables from file "Gains.dat". They were calculated by the Flixan Transient LQR program as already mentioned. The t_{go} is calculated from the relative axial position, velocity and acceleration and used to look-up the gains which increase as t_{go} gets shorter resulting into an exponentially increasing control bandwidth.



The Kalman-Filter and the control system are separate subsystems in this simulation. The Kalman-Filter is now used to estimate the four state variables from the relative position and the accelerometer measurements. The estimated states are multiplied with the four time varying gains to produce the acceleration command. The gains are functions of t_{go} which is calculated from the relative x-axis acceleration, velocity and range to go.

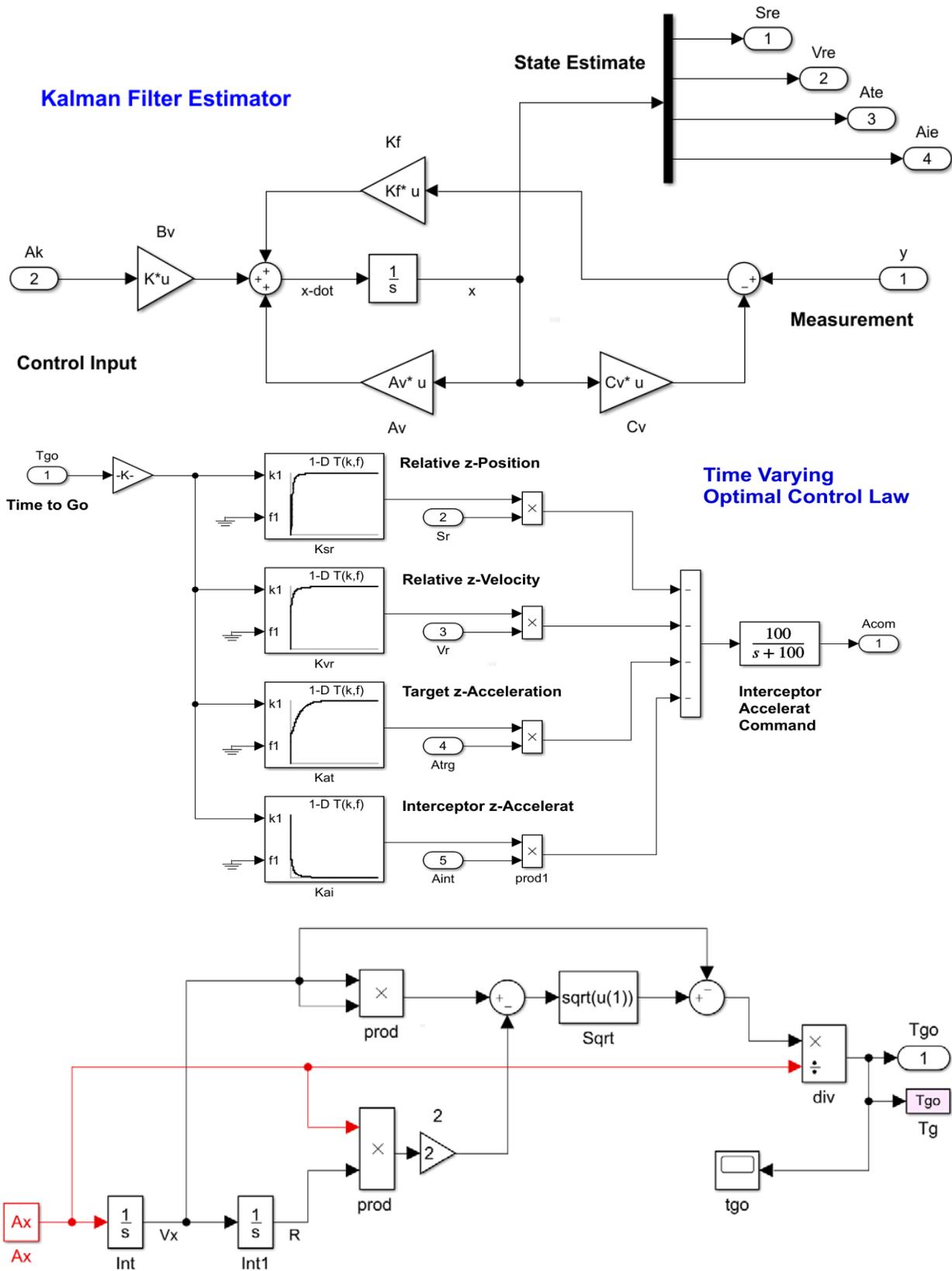


Figure 6.2.7 Simulation Model "EndGame_Sim2s.mdl" that uses Time-Varying Gains

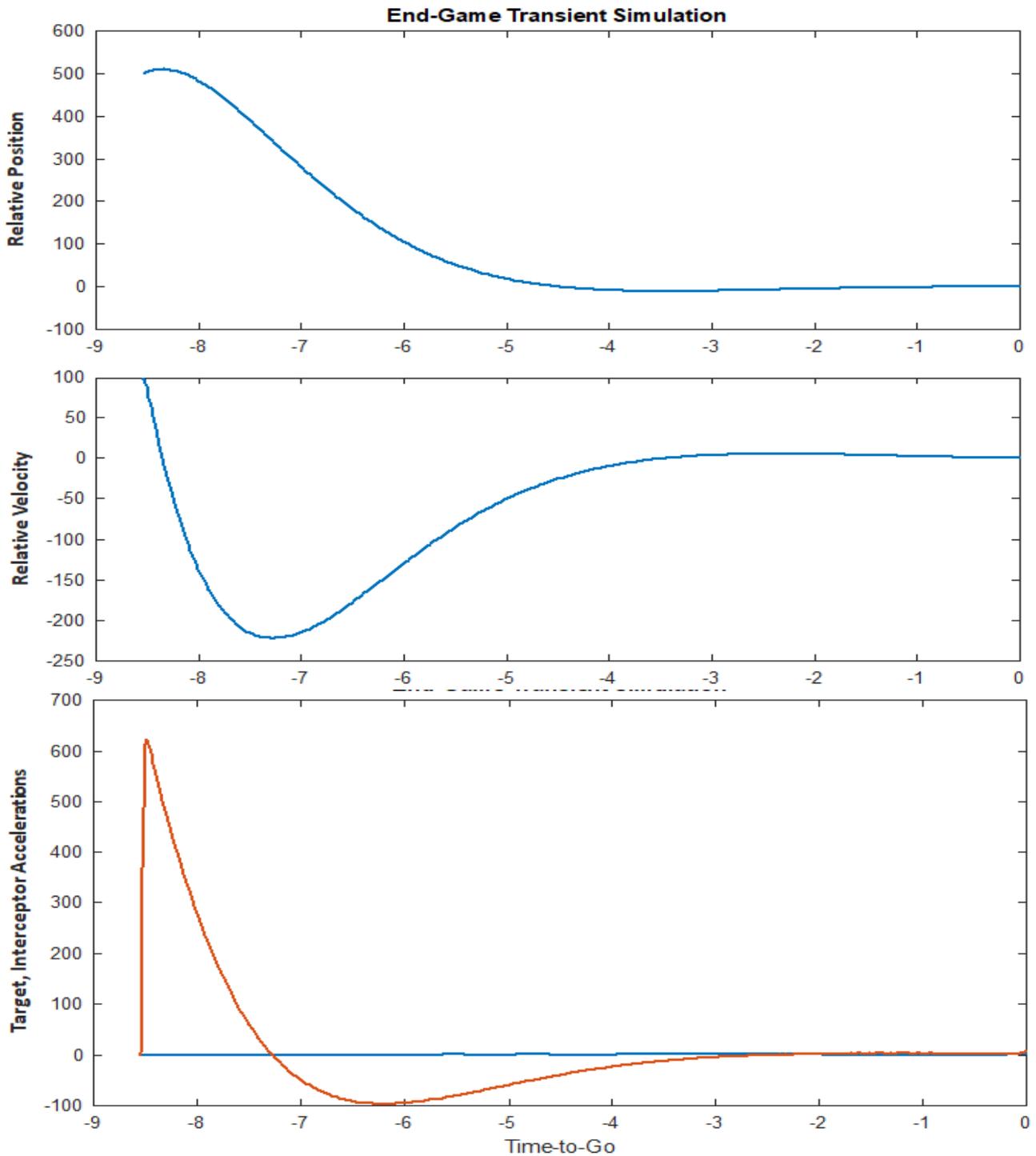


Figure 6.2.8 System's Response from Non-Zero Initial Conditions of Relative Position and Velocity

Figure 6.2.8 shows the response of the time-varying control system to non-zero initial conditions versus time-to-go, beginning 8.5 sec before impact when the gains are still at steady state. Beginning with initial position and velocity errors 500 (feet) and 100 (ft/sec) respectively, the interceptor (orange) accelerates in order to bring the final position and velocity errors very close to zero at impact.

6.2.5. Discrete-Time Analysis

The discrete-time analysis is similar. The files are located in subfolder “LQG\Examples\End-Game\Discrete”. It includes the input file “End_Game_z.Inp”, shown below, that contains input data for the design programs. That is, for the discrete steady-state LQR control, the discrete time-varying state-feedback LQR, the discrete Kalman-Filter, and for the discrete steady-state dynamic output-feedback LQG design. The input file also includes a batch set for fast data processing and datasets for Matlab conversions. The systems filename “End_Game_z.Qdr” contains the End-Game dynamic model “Simple End-Game Model, Z-Transform”, which is a z-transformation of the continuous model “Simple End-Game Model” described in Equation 6.2.2, discretized at 40 (msec) sampling period. The systems file also contains the control design matrices: Q_c , R_c , and P_1 , the Kalman-Filter design matrices Q_{mn} , R_{mn} , the control gain K_c and the Kalman-Filter gain K_f which are generated by the Flixan LQR design program.

```
BATCH MODE INSTRUCTIONS .....
Batch for preparing End-Game Control design Models
! This batch Generates LQR State-Feedback, Kalman-Filter and Output
! Feedback Dynamic Controller
Retain System      : Simple End-Game Model
Retain Matrix      : State Weight Matrix Qc (4x4)
Retain Matrix      : Control Weight Matrix Rc
Retain Matrix      : Terminal State Weight Matrix P1 (4x4)
Retain Matrix      : Measurement Noise Matrix Rmn (2x2)
Retain Matrix      : Process Noise Matrix Qpn (4x4)
Retain Matrix      : Input Noise Matrix G
Retain Matrix      : Process Noise Matrix Qpn1
!
S-Z-Transform      : Simple End-Game Model, Z-Transform
LQR Control Des    : LQR Control Design for Discrete End-Game Model
State Estimator    : Kalman-Filter Design 2 for Discrete End-Game Model
LQG Control Des    : LQG Control Design for Discrete End-Game Model
Transient LQR      : Transient LQR Design for Discrete End-Game Model
!
To Matlab Format    : Simple End-Game Model, Z-Transform
To Matlab Format    : LQG Control Design for Discrete End-Game Model
To Matlab Format    : LQR State-Feedback Control for Discrete End-Game Model
To Matlab Format    : Kalman-Filter Estimator 2 for Discrete End-Game Model
-----
TRANSFORM A SYSTEM (S-Z-W) ..... (Z system title, Comments, S-System title, Transform)
Simple End-Game Model, Z-Transform
! Discretize the Continuous End-Game Model at dT=0.04 sec Using the
! S to Z Transformation
!
Simple End-Game Model
From S-plane to Z-plane using the Z-Transform, dT= 0.04
-----
LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design for Discrete End-Game Model
! Design the Discrete Steady-State-Feedback Matrix Kc using the
! Output Criteria Matrix C= Identity
!
Plant Model Used to Design the Control System from:      Simple End-Game Model, Z-Transform
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix:      Qc4
Control Penalty Weight (Rc) is Matrix:      Rc
Discrete LQR Solution Using Asymptotic Method
LQR State-Feedback Control Gain Matrix Kc      LQR State-Feedback Control for Discrete End-
-----
```

KALMAN-BUCY FILTER STATE ESTIMATOR DESIGN

Kalman-Filter Design for Discrete End-Game Model

! Design the Discrete Kalman-Filter Gain Matrix Kf using the
! Process Noise Matrix G

! Plant Model Used to Design the Kalman-Filter from: Simple End-Game Model, Z-Transform
Input Process Noise Matrix is Matrix G Input Noise Matrix G
Process Noise Covariance Qpn is Matrix Qpn Process Noise Matrix Qpn1
Measurement Noise Covariance is Matrix Rmn Measurement Noise Matrix Rmn (2x2)
Kalman-Filter Estimator is Gain Matrix Kf Kalman-Filter Estimator for Discrete End-

KALMAN-BUCY FILTER STATE ESTIMATOR DESIGN

Kalman-Filter Design 2 for Discrete End-Game Model

! Plant Model Used to Design the Kalman-Filter from: Simple End-Game Model, Z-Transform
Input Process Noise Matrix (G) is the Identity Input Noise Matrix G
Process Noise Covariance Qpn is Matrix Qpn Process Noise Matrix Qpn (4x4)
Measurement Noise Covariance is Matrix Rmn Measurement Noise Matrix Rmn (2x2)
Kalman-Filter Estimator is Gain Matrix Kf Kalman-Filter Estimator 2 for Discrete End-

DYNAMIC OUTPUT FEEDBACK LQG CONTROL DESIGN

LQG Control Design for Discrete End-Game Model

! Combine State-Feedback with KF Gain to Design a Linear Quadratic
! Gaussian Control System for the Plant: Simple End-Game Model, Z-Transform

! Plant Model Used to Design the Control System from: Simple End-Game Model, Z-Transform
State-Feedback (Kc) is Gain Matrix : Kc LQR State-Feedback Control for Discrete End-
Game Model
Kalman-Filter Estim Kf is Gain Matrix: Kf Kalman-Filter Estimator 2 for Discrete End-
Game Model

TRANSIENT LQR CONTROL DESIGN WITH TIME-VARYING GAINS

Transient LQR Design for Discrete End-Game Model

! To Generate Time-Varying State-Feedback Gain Kc(t)
! as a Function of Time-to-Go, in File: Gains.dat

! Plant Model Used to Design the Control System from: Simple End-Game Model, Z-Transform
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix: Qc4 State Weight Matrix Qc (4x4)
Control Penalty Weight (Rc) is Matrix: Rc Control Weight Matrix Rc
Terminal State Penalty Weigh P1 Matrix P1 Terminal State Weight Matrix P1 (4x4)
Discrete LQR Solution, Final Time (Tf) in (sec) 20.0

CONVERT TO MATLAB FORMAT (Title, System/Matrix, m-filename)

Simple End-Game Model, Z-Transform

System
end_game

CONVERT TO MATLAB FORMAT (Title, System/Matrix, m-filename)

LQG Control Design for Discrete End-Game Model

System
Control

CONVERT TO MATLAB FORMAT (Title, System/Matrix, m-filename)

Kalman-Filter Estimator 2 for Discrete End-Game Model

Matrix Kf

CONVERT TO MATLAB FORMAT (Title, System/Matrix, m-filename)

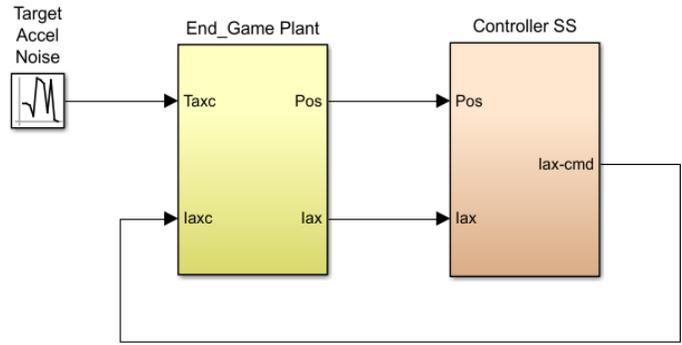
LQR State-Feedback Control for Discrete End-Game Model

Matrix Kc

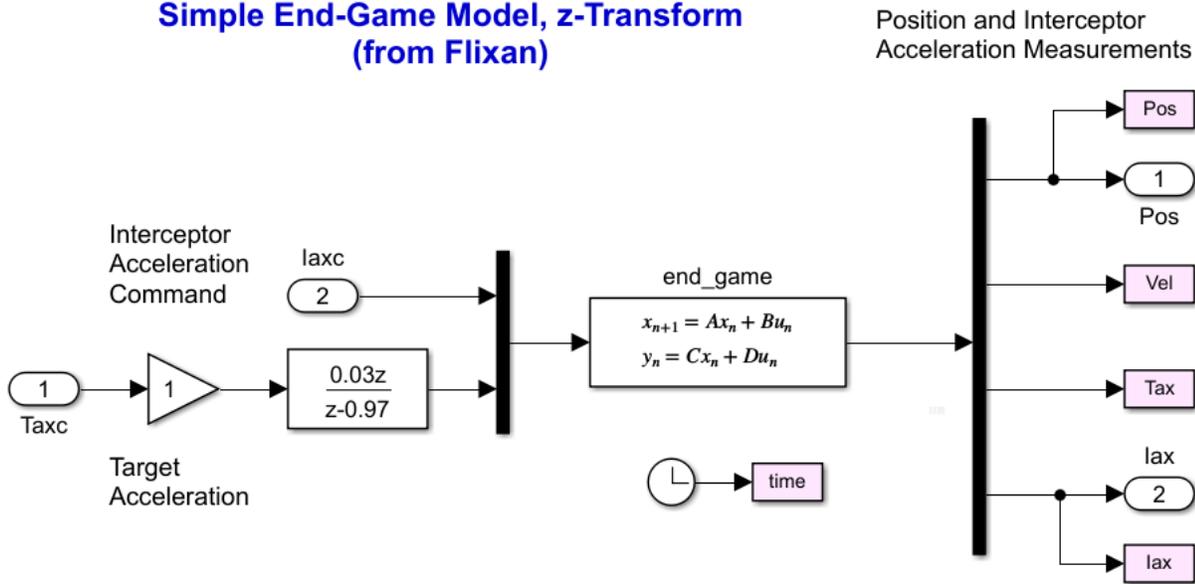
The discrete LQR transient design program also calculates the time varying state-feedback gain $K_c(t)$ as a function of time-to-go. It requires the weight matrices: Q_c , R_c , and P_1 , and the initial time-to-go (20 sec). The gains versus time are saved in file "Gains.dat" which is used as a look-up table in the simulation. The discrete end-game dynamic model is saved in file "end_game.m", and the discrete steady-state output-feedback controller is saved in "control.m" for Matlab analysis. The gain matrices K_c and K_f are also saved and loaded into Matlab. Two discrete Simulink models are included for analysis: a steady-state model, and a time-varying model, both running at 40 msec sampling.

6.2.5.1 Discrete Steady-State Simulation

The steady-state model is used to analyze the system when the target is sufficiently far away from the interceptor and the control gains are constant. The Simulink model is "EndGame-Sim1z.mdl", shown in Figure 6.2.9, and it is located in "LQG\ Examples\End-Game\ Discrete" subfolder.



Simple End-Game Model, z-Transform (from Flixan)



Steady-State Discrete Controller (from Flixan)

LQG Control Design for Discrete End-Game Model

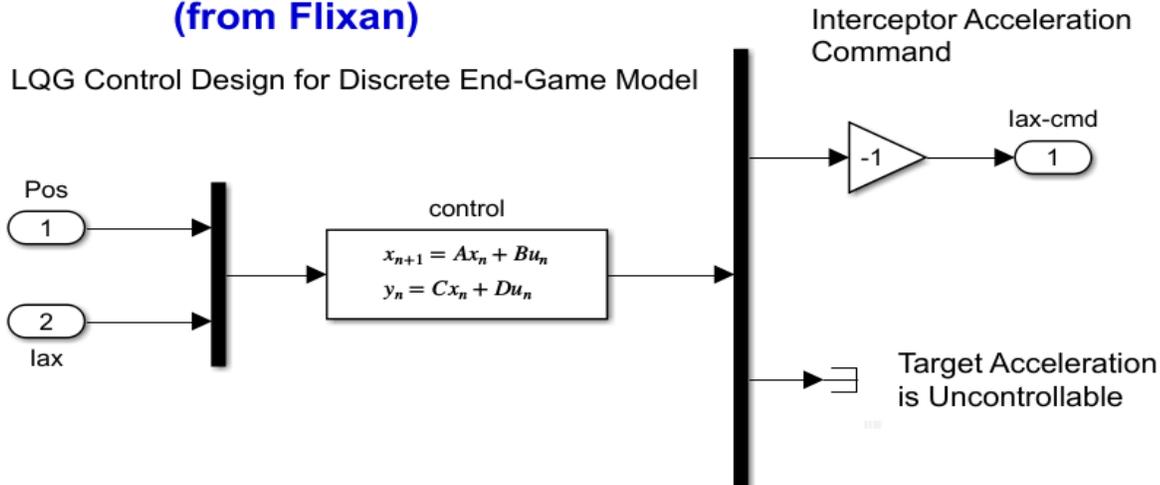


Figure 6.2.9 Discrete Steady-State Simulation Model "EndGame_Sim1z.mdl"

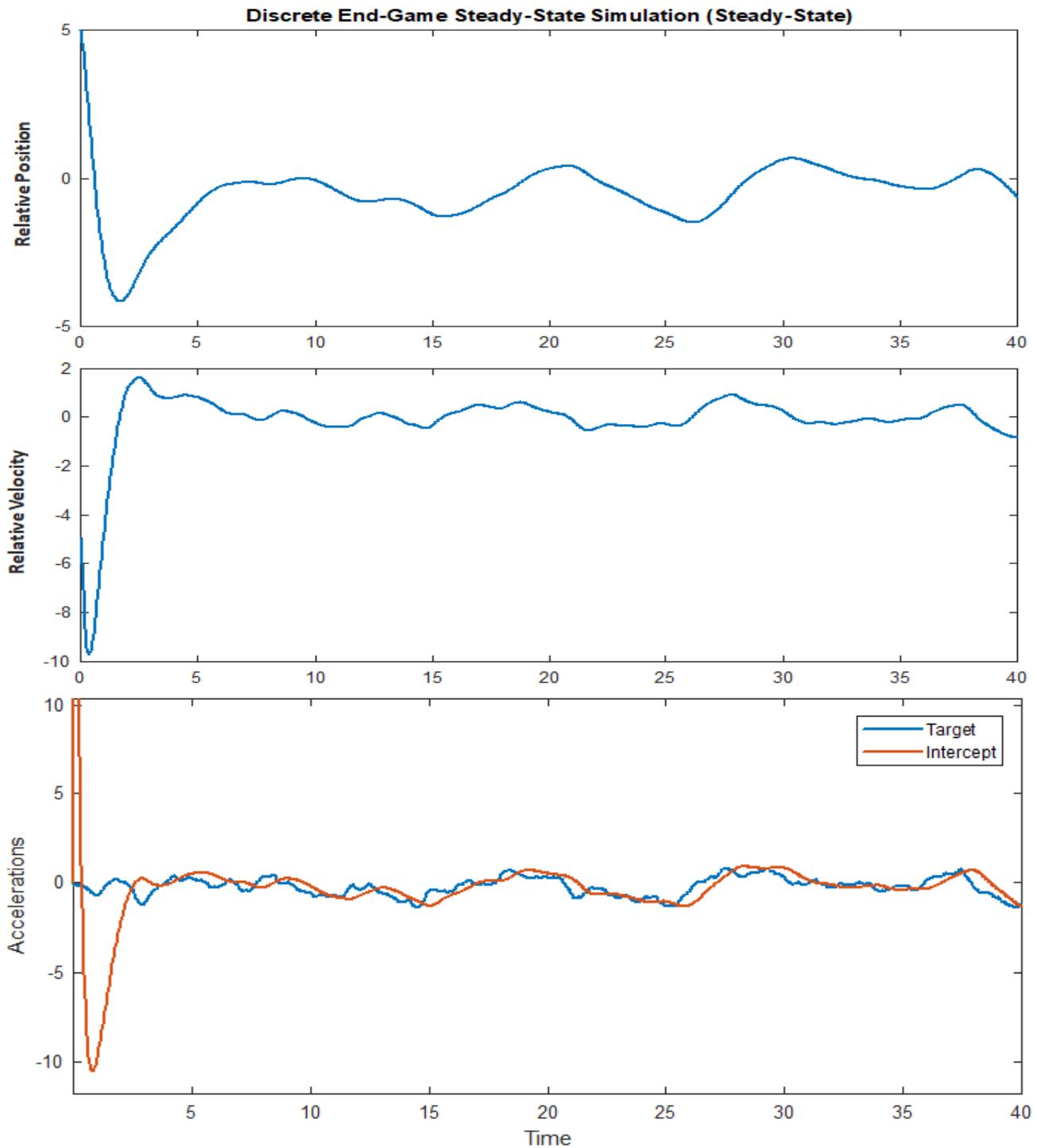


Figure 6.2.10 Discrete System's Response from Non-Zero Initial Conditions of Relative Position and Velocity

Figure 6.2.10 shows discrete system's response to a randomly accelerating target. The dynamic model is initialized at some arbitrary non-zero relative position and velocity errors. The interceptor acceleration responds to the target's average acceleration and the relative position and velocity are considerably reduced. Notice that this is steady-state condition where the interceptor's response is slow. The engagement becomes a lot more dynamic when t_{go} approaches zero.

**Simple End-Game Model, Z-Transform
(from Flixan)**

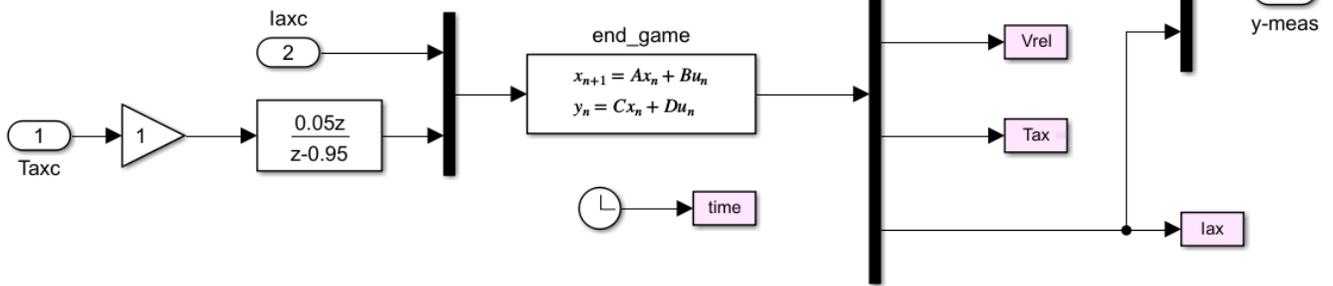


Figure 6.2.11 Discrete Simulation Model “EndGame_Sim2z.mdl” that uses Time-Varying Gains

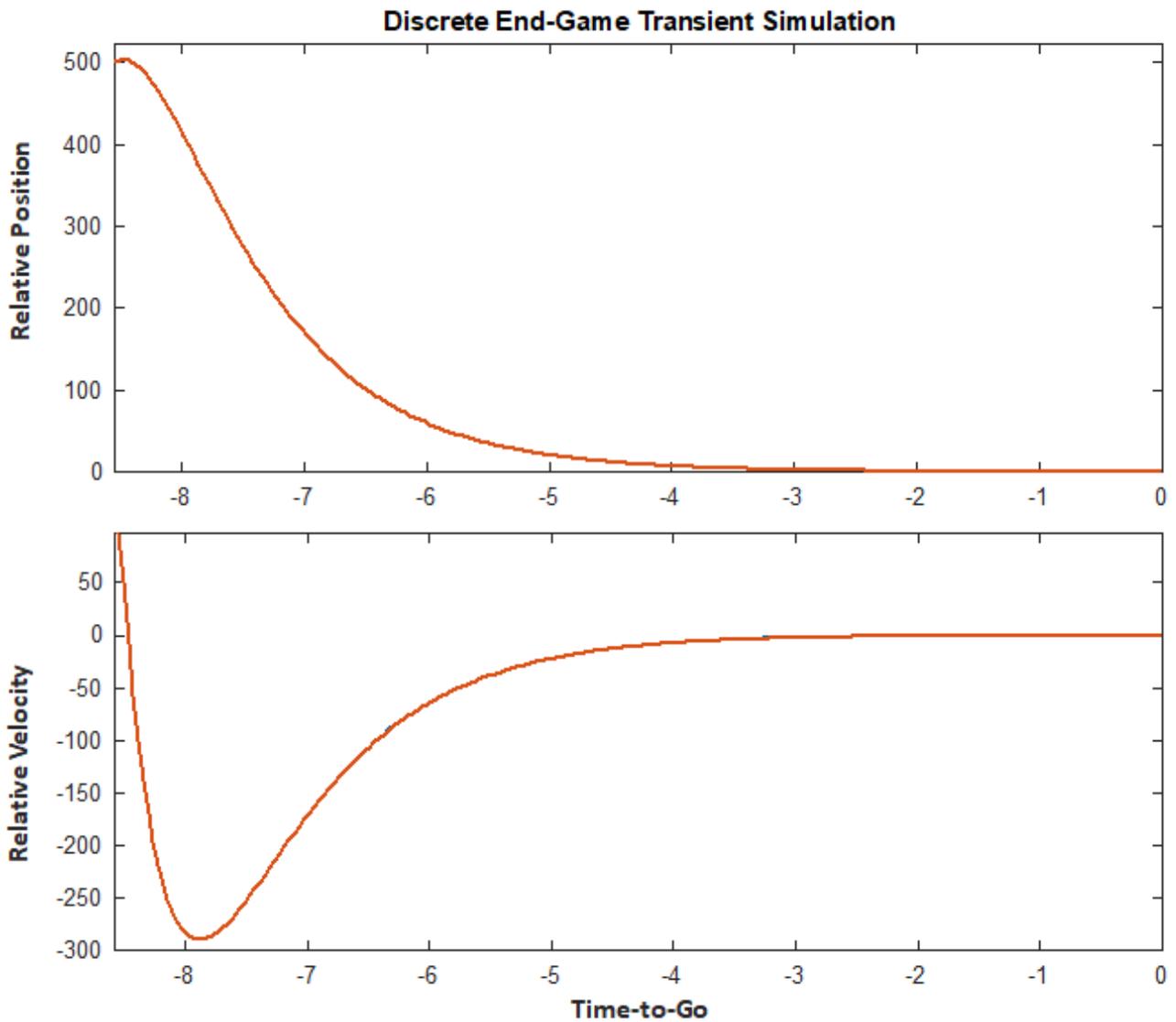
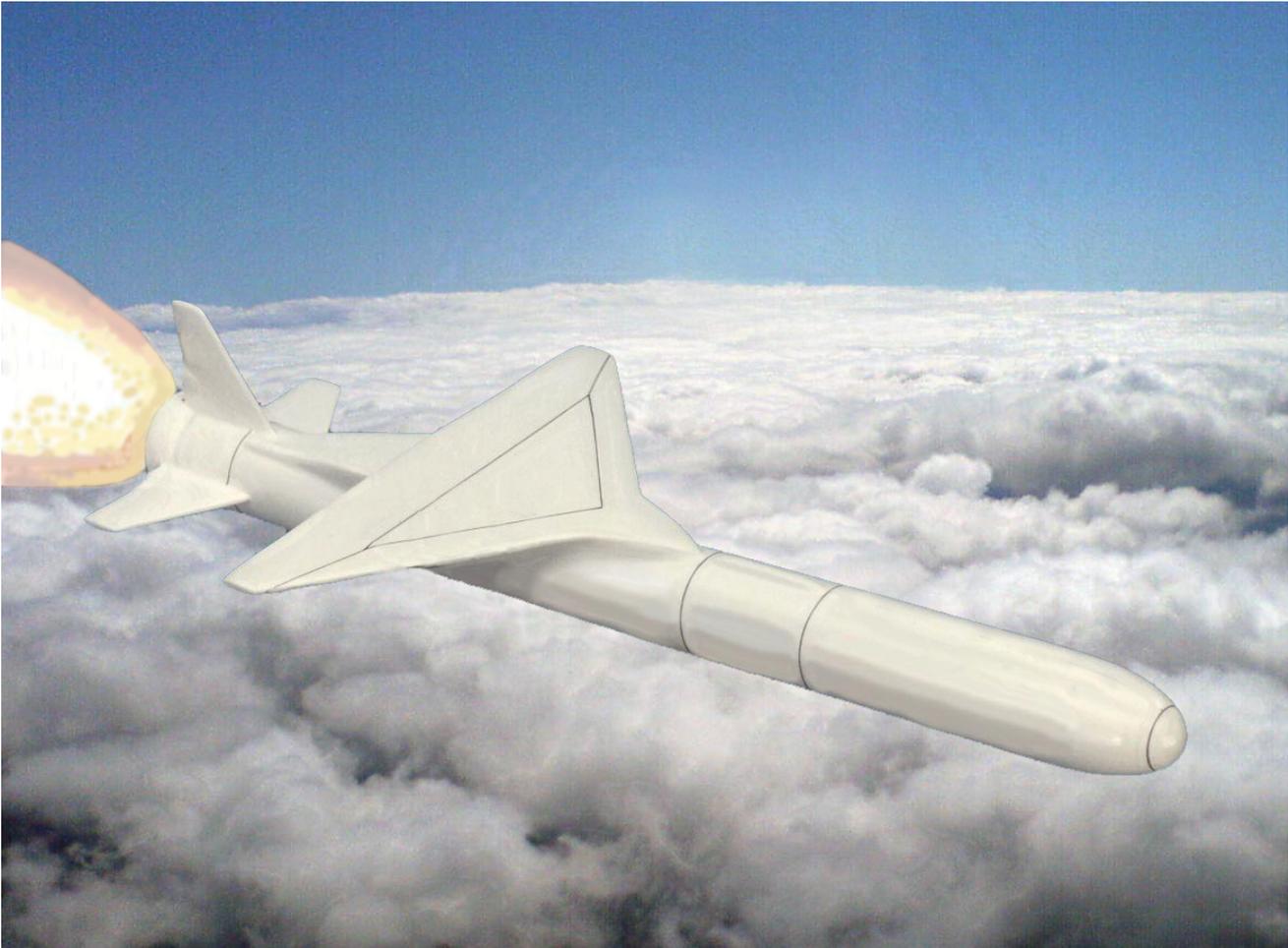


Figure 6.2.12 Discrete System’s Response from Non-Zero Initial Conditions of Relative Position and Velocity

Control Design of a Missile with Wing



In this example we analyze a cruising missile that has a small wing to provide lift and a fixed thrust engine that does not gimbal nor throttle. The missile is released horizontally from an aircraft and it climbs at high altitudes. It is controlled by three aero-surfaces located in the tail section consisting of: a vertical rudder mainly for yaw control and two horizontal rotating fins for pitch and roll control, see Figure 1.1. There are no control surfaces on the wing. Since the engine is not gimbaling, it is the wing in combination with the elevon aerosurfaces that provides the necessary lift for the vehicle to climb. The attitude, rate, and acceleration are measured by an Inertial Measurements Unit (IMU) located in the front section. The angles of attack and sideslip are not measured relative to the wind but the flight path angle γ and the heading direction ξ are inertially estimated from navigation. We will use Flixan to generate dynamic models at a critical flight condition, which is: Mach 2.5, 10 degrees of angle of attack, and high dynamic pressure of 1220 psf. We will design LQR control laws for the pitch and lateral dynamics separately, and analyze stability and performance using Matlab.

1. Flight Control System Description

Figure 1.1 shows the missile with the wing and the three tail aerosurfaces consisting of a vertical rudder for yaw control, an elevon for pitch control produced by equally rotating the left and right fins in the same direction, and an aileron for roll control produced by rotating the left and right fins differentially. The missile is released horizontally from an aircraft, climbs to orbital altitude and tracks a pre-calculated flight path and heading directions, mainly along the direction it is released.

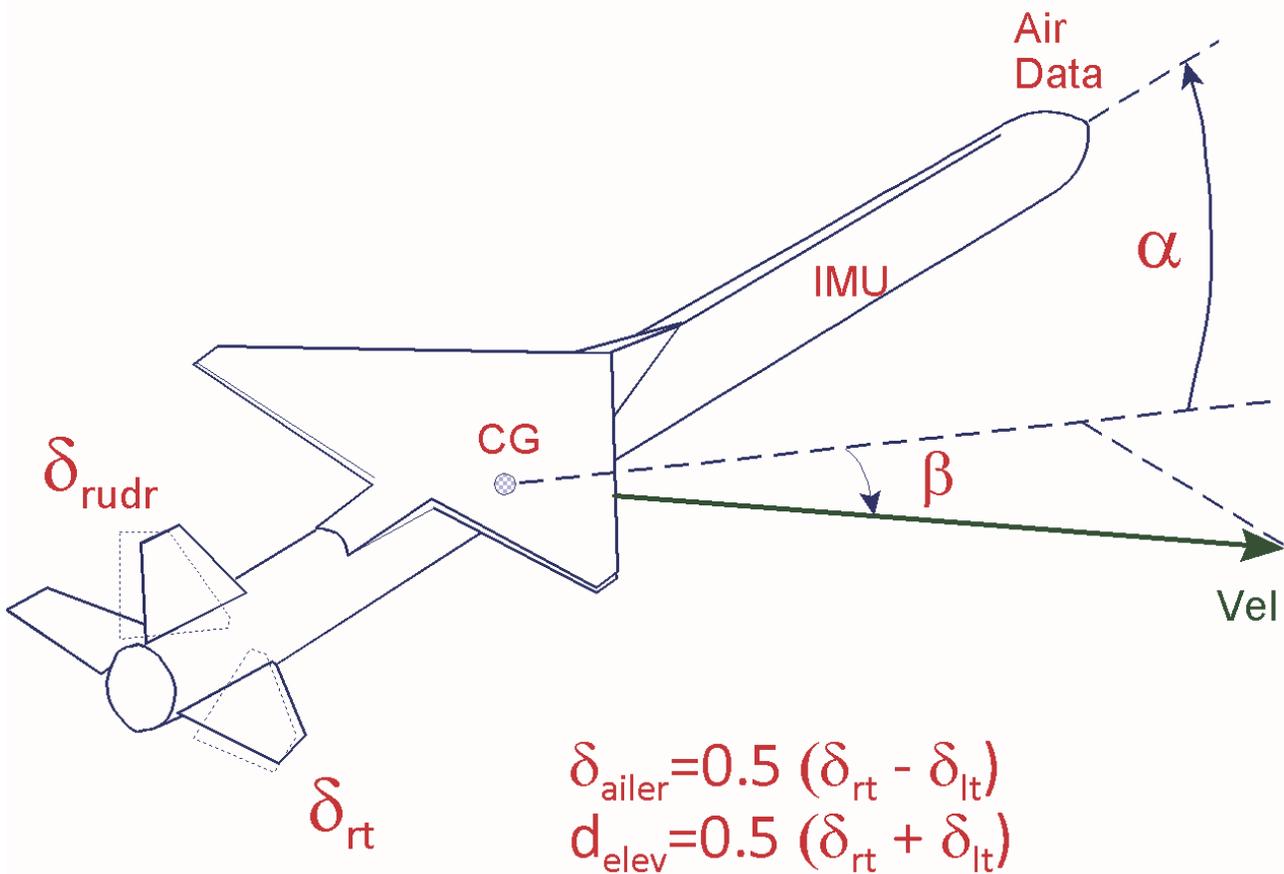


Figure 1.1 Missile Configuration showing the Aerosurfaces, Wing, CG, and Sensor Locations.

The thrust is not used for control but it produces an acceleration which is captured in the vehicle data and model. The purpose of the flight control system is to stabilize the vehicle and to track a pre-designed trajectory path in both: longitudinal gamma-tracking, and in the lateral heading direction tracking. Since the vehicle is perfectly symmetric the analysis will be separated in pitch and lateral control design and analysis that will be performed in separate subdirectories. The pitch vehicle model is in the input file "Pitch_LQR_Des.Inp" which is located in directory "Flixan\Control Analysis\LQG\Examples\Missile Control Design\Pitch LQR". The lateral vehicle model is in the input file "Later_LQR_Des.Inp" which is located in directory "Flixan\Control Analysis\LQG\Examples\Missile Control Design\Lateral LQR". Pitch and Lateral control design models will be created for LQR state-feedback and we assume that all states \underline{x} are available for feedback.

The design models will be augmented by including the aerosurface actuators and integrals of some of the states. The augmented design models improve speed of response and the tracking performance of the control system. We will also create pitch and lateral models for control analysis and simulations. The Flixan program will be used to perform dynamic modeling and control design and Matlab for the simulations. The dynamic models and control gains are converted from the system files and loaded into Matlab for analysis.

Note that in this example the incidence angles α and β , which are used to synthesize the flight-path and heading directions, do not see the effects of a wind-gust directly because γ and ξ are estimated from navigation and they do not represent motion relative to the moving air mass. This is introduced in the flight vehicle input data by a flag label **"NoWind Alpha"** in the flags line, to indicate the type of (α, β) measurement. It means that the wind velocity components w_{gust} and v_{gust} are not included in the α and β calculations, only the vehicle velocities w and v . A wind-gust, however, will produce forces and moments on the vehicle and it will affect its motion, but the gust itself is not seen directly in the output as it would be if it was an air-data probe, only its effect on the vehicle will be observable.

2.1 Longitudinal Control Design and Analysis

The input file for the longitudinal axis design is “*Pitch_LQR_Des.Inp*” located in subdirectory “*Control Analysis\LQG\Examples\Missile Control Design\Pitch LQR*”. It contains several Flixan datasets that generate plant models and perform steady-state LQR state-feedback control design. They are processed by a batch set located at the top of the file. The batch first retains the control weight matrices Q_c and R_c from getting erased in systems file “*Pitch_LQR_Des.Qdr*”. Then it generates the vehicle model “*Missile with Wing, Mach: 2.5, Qbar: 1220*” that includes both pitch and lateral dynamics. The initial pitch design model is then extracted from the above system and saved as “*Missile with Wing Pitch Design Model*”. It consists of one input, Elevon deflection in (rad), and 3 outputs: pitch attitude, rate, and angle of attack in radians. A second longitudinal system is also created with title: “*Missile with Wing Pitch Analysis Model*”. It includes a wind-gust velocity input in (feet/sec) and other outputs, and it will be used in simulations. The direction of the gust is perpendicular to the vehicle x-axis, and along the $-z$ direction to excite the pitch dynamics, as defined in the vehicle input data by the wind azimuth and elevation angles (0° and 90°).

```
BATCH MODE INSTRUCTIONS .....
Batch for Designing Missile with Wing Pitch Models and Gains
!
! This batch set creates the Design and Analysis models for a
! Missile with Wing at 2.5 Mach, and performs LQR design.
! The Missile has a fixed Thrust and it is controlled by 3 Aerosurfaces
!
!           Control design Matrices
Retain Matrix   : State Weight Matrix Qc (5x5)
Retain Matrix   : Control Weight Matrix Rc
!
Flight Vehicle  : Missile with Wing, Mach: 2.5, Qbar: 1220
System Modificat : Missile with Wing Pitch Design Model
System Modificat : Missile with Wing Pitch Analysis Model
Transf-Functions : Actuator: 34/(s+34)
Transf-Functions : Integrator
System Connection: Augmented Pitch Design Model
System Modificat : Augmented Pitch Design Model-2
LQR Control Des  : LQR Control Design for Augmented Design Model
!
!           Convert to Matlab
To Matlab Format : LQR State-Feedback Control for Augmented Design Model
To Matlab Format : Missile with Wing Pitch Analysis Model
-----
```

FLIGHT VEHICLE INPUT DATA

Missile with Wing, Mach: 2.5, Qbar: 1220

**! Rigid-Body Missile controlled by 3 aerosurfaces. The engine has fixed thrust
! and does not gimbal**

Body Axes Output, Attitude=Euler Angles, NoWind Alpha

Vehicle Mass (lb-sec²/ft), Gravity Accelerat. (g) (ft/sec²), Earth Radius (Re) (ft) : 1219.1, 32.07,
Moments and products of Inertias Ixx, Iyy, Izz, Ixy, Ixz, Iyz, in (lb-sec²-ft) : 0.4063E+04 0.1654E+06
CG location with respect to the Vehicle Reference Point, Xcg, Ycg, Zcg, in (feet) : 26.19, 0.0, -0.15
Vehicle Mach Number, Velocity Vo (ft/sec), Dynamic Pressure (psf), Altitude (feet) : 2.5, 2427.4,1220.6,
Inertial Acceleration Vo_dot, Sensed Body Axes Accelerations Ax,Ay,Az (ft/sec²) : 60.0, 60.0, 0.0, 10.5
Angles of Attack and Sideslip (deg), alpha, beta rates (deg/sec) : 10.5, 0.0, 0.0, 0.0
Vehicle Attitude Euler Angles, Phi_o,Thet_o,Psi_o (deg), Body Rates Po,Qo,Ro (deg/sec) : 0.0,39.6,0.0, 0.0, 0.132,
Wind Gust Vel wrt Vehi (Azim & Elev) angles (deg), or Force(lb), Torque(ft-lb), locat:xyz: Gust 00.0 90.0
Surface Reference Area (feet²), Mean Aerodynamic Chord (ft), Wing Span in (feet) : 145.4, 22.0, 22.0
Aero Moment Reference Center (Xmrc,Ymrc,Zmrc) Location in (ft), {Partial_rho/ Partial_H} : 26.19, 0.0, -0.238, 0.0
Aero Force Coef/Deriv (1/deg), Along -X, {Cao,Ca_alf,PCa/PV,PCa/Ph,Ca_alfdot,Ca_q,Ca_bet} : 0.1, 0.002, 0.0, 0.0,
Aero Force Coeffic/Derivat (1/deg), Along Y, {Cyo,Cy_bet,Cy_r,Cy_alf,Cy_p,Cy_betdot,Cy_V} : 0.0, -0.023, 0.0, 0.0,
Aero Force Coeff/Deriv (1/deg), Along Z, {Czo,Cz_alf,Cz_q,Cz_bet,PCz/Ph,Cz_alfdot,PCz/PV} : -0.1, -0.032, 0.0, 0.0,
Aero Moment Coeffic/Derivat (1/deg), Roll: {Clo, Cl_beta, Cl_betdot, Cl_p, Cl_r, Cl_alfa} : 0.0, -0.0017,0.0,-0.243,
Aero Moment Coeff/Deriv (1/deg), Pitch: {Cmo,Cm_alfa,Cm_alfdot,Cm_bet,Cm_q,PCm/PV,PCm/Ph} : -0.037,-0.011, 0.0,0.0,
Aero Moment Coeffic/Derivat (1/deg), Yaw : {Cno, Cn_beta, Cn_betdot, Cn_p, Cn_r, Cn_alfa} : 0.0, 5.6e-4, 0.0,0.1388,

Number of Control Surfaces, With or No TWD (Tail-Wags-Dog and Hinge Moment Dynamics) ? : 3 No TWD

Control Surface No: 1 Elevator
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.000
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00003 -0.0 -0.00087, 0.00
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : 0.0 -0.0072 0.0 0.0

Control Surface No: 2 Aileron
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.0
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00003 0.0011 0.0 0.0
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : -6.54e-4 0.0 -0.0014 0.0

Control Surface No: 3 Rudder
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.000
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00001 0.0034 0.0 0.0
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : 5.9456e-4 0.0 -0.0035

Number of Bending Modes : 0

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Missile with Wing Pitch Design Model

Missile with Wing, Mach: 2.5, Qbar: 1220

! The initial pitch design system is extracted from the coupled RB system above

!

TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract Inputs : 1
Extract States : 3 4 7
Extract Outputs: 3 4 7

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Missile with Wing Pitch Analysis Model

Missile with Wing, Mach: 2.5, Qbar: 1220

! The Pitch Analysis/ Simulation system is extracted from the coupled RB system above

!

TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract Inputs : 1 4
Extract States : 3 4 7 9 10
Extract Outputs: 3 4 7 9 10 14

The system modification datasets extract the longitudinal variables from the coupled system "Missile with Wing, Mach: 2.5, Qbar: 1220" and save them in file "Pitch_LQR_Des.Qdr" as separate systems.

The original design plant, however, is not capable to produce an efficient control design. In the longitudinal direction we would like to command and track a pre-calculated flight path angle (γ) and the initial design model is not equipped to regulate γ . We must create, therefore, and regulate a “ γ -integral” state and include it in the design model. It is also good to include a simple actuator model in the plant dynamics because it introduces more plant information in the design and makes the control system more efficient with less phase-lag. The two additional variables γ and δ_{elevon} are both measurable.

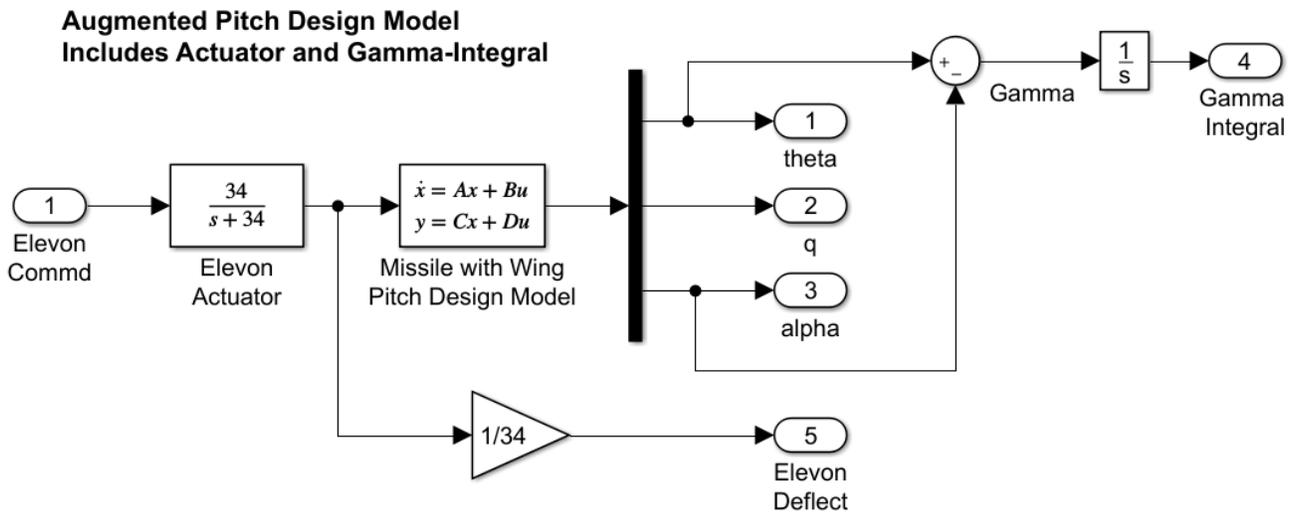


Figure 2.1 Augmented Longitudinal Design Plant for LQR Control Design

Figure 2.1 shows the augmented plant for the longitudinal LQR control design. The following interconnection dataset combines the 3 subsystems and generates the augmented system as “*Augmented Pitch Design Model*”. The order of the states, however, is not the same as the outputs and it is modified for convenience to “*Augmented Pitch Design Model-2*” which makes the C matrix equal to the identity I_5 .

INTERCONNECTION OF SYSTEMS

Augmented Pitch Design Model

! Create a 5-State Augmented Model that Includes Gamma-integral and

! Elevon deflection in the state vector for Pitch Control Design

!

Titles of Systems to be Combined

Title 1 Actuator: 34/(s+34)

Title 2 Missile with Wing Pitch Design Model

Title 3 Integrator

SYSTEM INPUTS TO SUBSYSTEM 1

System Input 1 to Subsystem 1, Input 1, Gain= 1.0

to Actuator
Delta Command

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 2

System Output 1 from Subsystem 2, Output 1, Gain= 1.0

System Output 2 from Subsystem 2, Output 2, Gain= 1.0

System Output 3 from Subsystem 2, Output 3, Gain= 1.0

Vehicle Plant
theta
q - pitch rate
alpha

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 3

System Output 4 from Subsystem 3, Output 1, Gain= 1.0

Integrator
gamma-integral

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 1

System Output 5 from Subsystem 1, Output 1, Gain= 0.0294118

Actuator
delta-elevon

.....

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 2

Subsystem 1, Output 1 to Subsystem 2, Input 1, Gain= 1.0000

Actuator to Vehicle
Elevon deflect

.....

SUBSYSTEM NO 2 GOES TO SUBSYSTEM NO 3

Subsystem 2, Output 1 to Subsystem 3, Input 1, Gain= 1.0000

Subsystem 2, Output 3 to Subsystem 3, Input 1, Gain= -1.0000

Vehicle to Integrator
Gamma= Theta
-Alpha

.....

Definitions of Inputs = 1

Elevon Deflection Command (delta) rad

Definitions of Outputs = 5

Pitch Attitude, theta (rad)

Pitch Rate, q (rad/sec)

Angle of Attack, alpha (rad)

Gamma-Integral (rad-sec)

Elevon Deflection, delta-elev (rad)

SYSTEM OF TRANSFER FUNCTIONS ...

Actuator: 34/(s+34)

! First order Actuator 34 (rad/sec) Bandwidth

Continuous

TF. Block # 1 34/(s+34)

Order of Numer, Denom= 0 1

Numer 0.0 34.0

Denom 1.0 34.0

Block #, from Input #, Gain

1 1 1.00000

.....

Outpt #, from Block #, Gain

1 1 1.00000

.....

Definitions of Inputs = 1

Delta Command

Definitions of Outputs = 1

Delta Out

SYSTEM OF TRANSFER FUNCTIONS ...

Integrator

Continuous

TF. Block # 1 1/s

Order of Numer, Denom= 0 1

Numer 0.0 1.0

Denom 1.0 0.0

Block #, from Input #, Gain

1 1 1.00000

.....

Outpt #, from Block #, Gain

1 1 1.00000

.....

```

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)
Augmented Pitch Design Model-2
Augmented Pitch Design Model
! Rearrange the Order of States to be the same as the Outputs
! Makes C=Identity
TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS
Extract States :   2   3   4   5   1
Extract Outputs:   1   2   3   4   5
-----

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design for Augmented Design Model
Plant Model Used to Design the Control System from:      Augmented Pitch Design Model-2
Criteria Optimization Output is Matrix C
State Penalty Weight (Qc) is Matrix:   Qc5              State Weight Matrix Qc (5x5)
Control Penalty Weight (Rc) is Matrix: Rc              Control Weight Matrix Rc
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kc              LQR State-Feedback Control for Augmented Design Model
-----

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Control for Augmented Design Model
Matrix Kc
-----

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Missile with Wing Pitch Analysis Model
System
Vehi_pitch.m
-----

END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-

```

The dataset “*LQR Control Design for Augmented Design Model*” performs the LQR control design on the plant “Augmented Pitch Design Model-2”. It uses the C matrix for criteria which is the identity matrix and the (5x5) weight matrix Q_c to penalize the states individually. The scalar R_c penalizes the Elevon control. The weight matrices are already set in the systems file. The LQR program generates the (1x5) state-feedback matrix K_c that stabilizes the plant by closing the control loop between the state vector and the Elevon input. The matrix is also saved in the systems file under the title “*LQR State-Feedback Control for Augmented Design Model*”. The matrix K_c and the pitch analysis model are also saved in Matlab format as “Kc.mat” and “vehi_pitch.m” respectively for further analysis.

2.2 Longitudinal Simulation

The simulation model “*Pitch_Sim.mdl*” in Figure 2.2, is in folder “*Flixan\Control Analysis\LQG\Examples\Missile Control Design\Pitch LQR*”. It has the 5-state-feedback loop closed via matrix K_c which includes γ -integral and δ_{elevon} feedback in addition to the feedback from the original vehicle states (θ, q, α). In Figure 2.3 the vehicle is commanded to perform 1° increase in γ . In Figure 2.4 the missile is excited by an upward wind-gust velocity impulse.

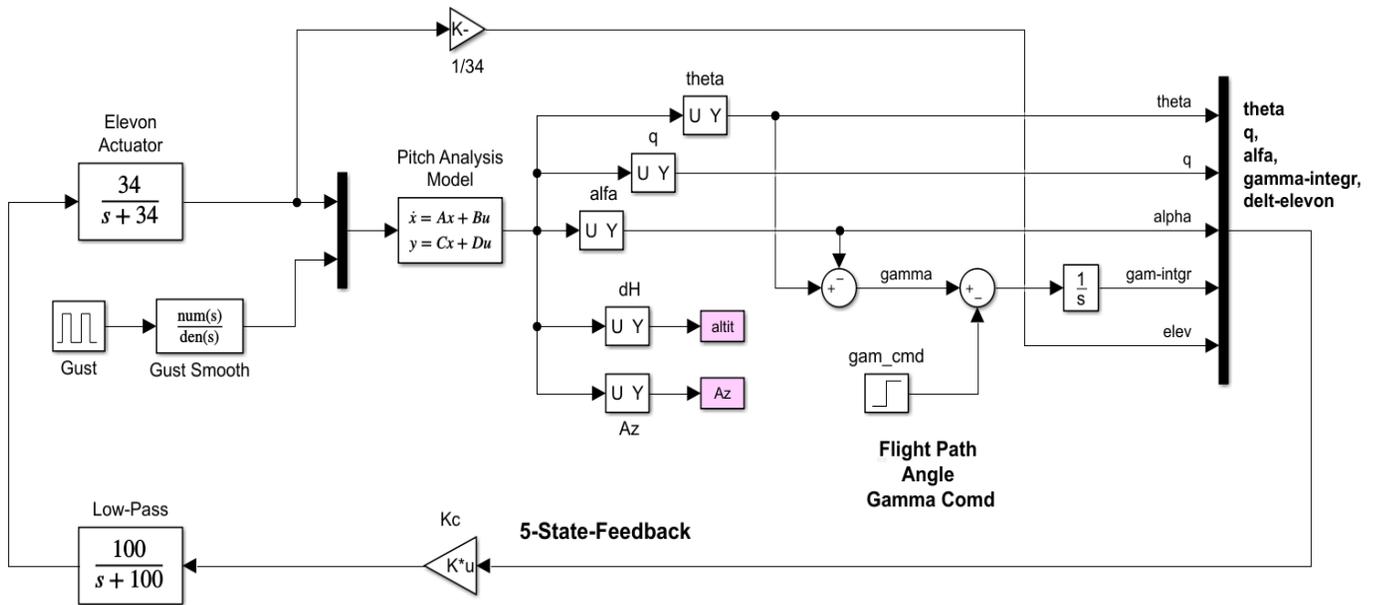


Figure 2.2 Longitudinal Axes Closed-Loop Simulation Model "Pitch_Sim.mdl"

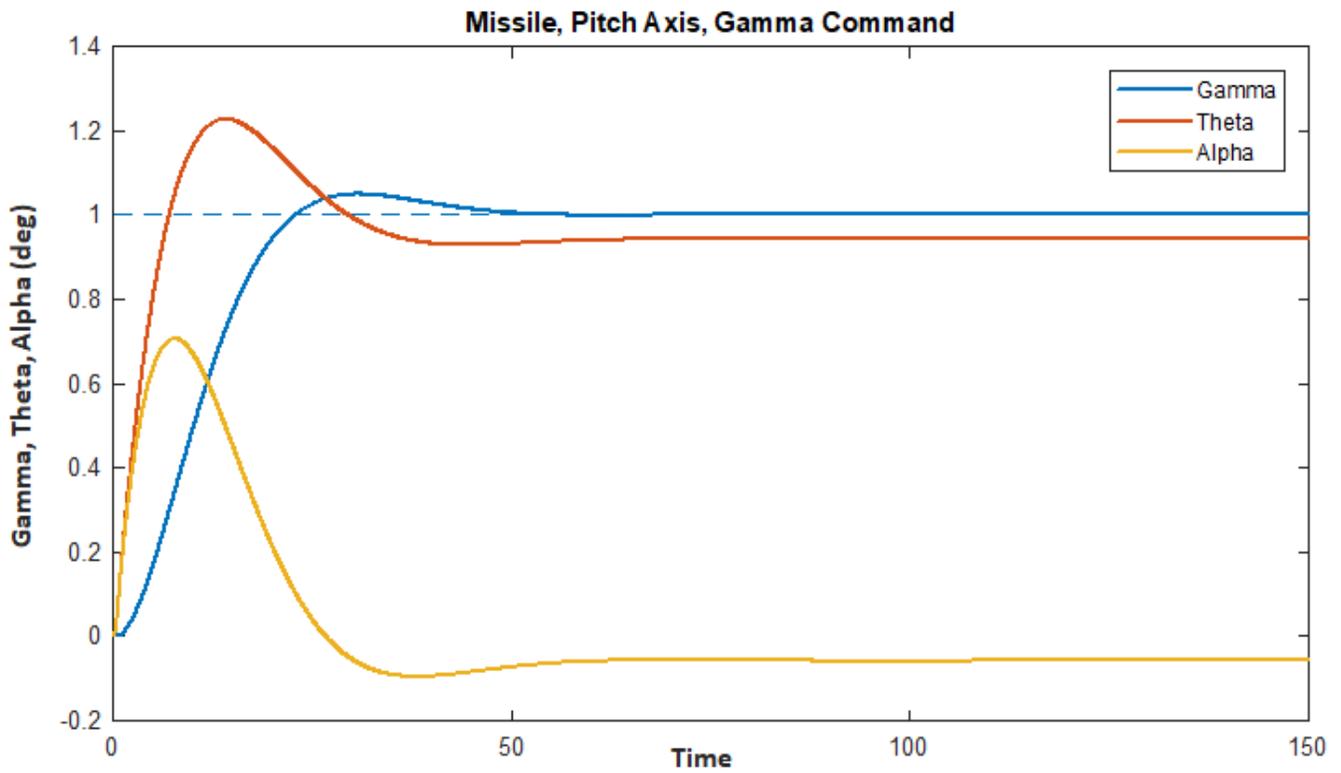


Figure 2.3 Flight Path Angle Responds to 1 degree Gamma Command

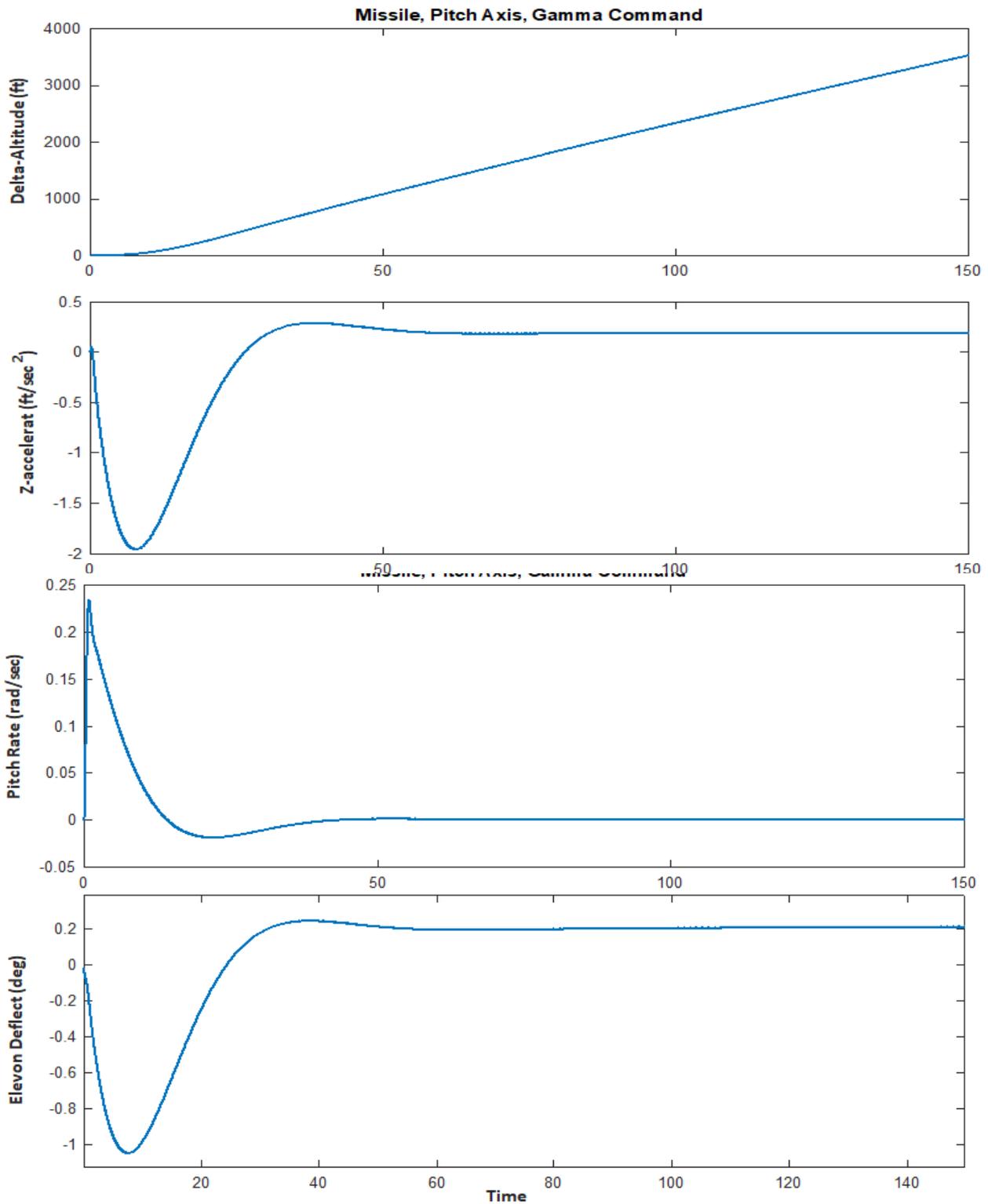


Figure 2.3b Missile Responds to the Gamma Command. Negative Elevon produces a Positive Pitch Rate, Negative (upwards) Acceleration and the Missile is Steadily Climbing at Higher Altitudes

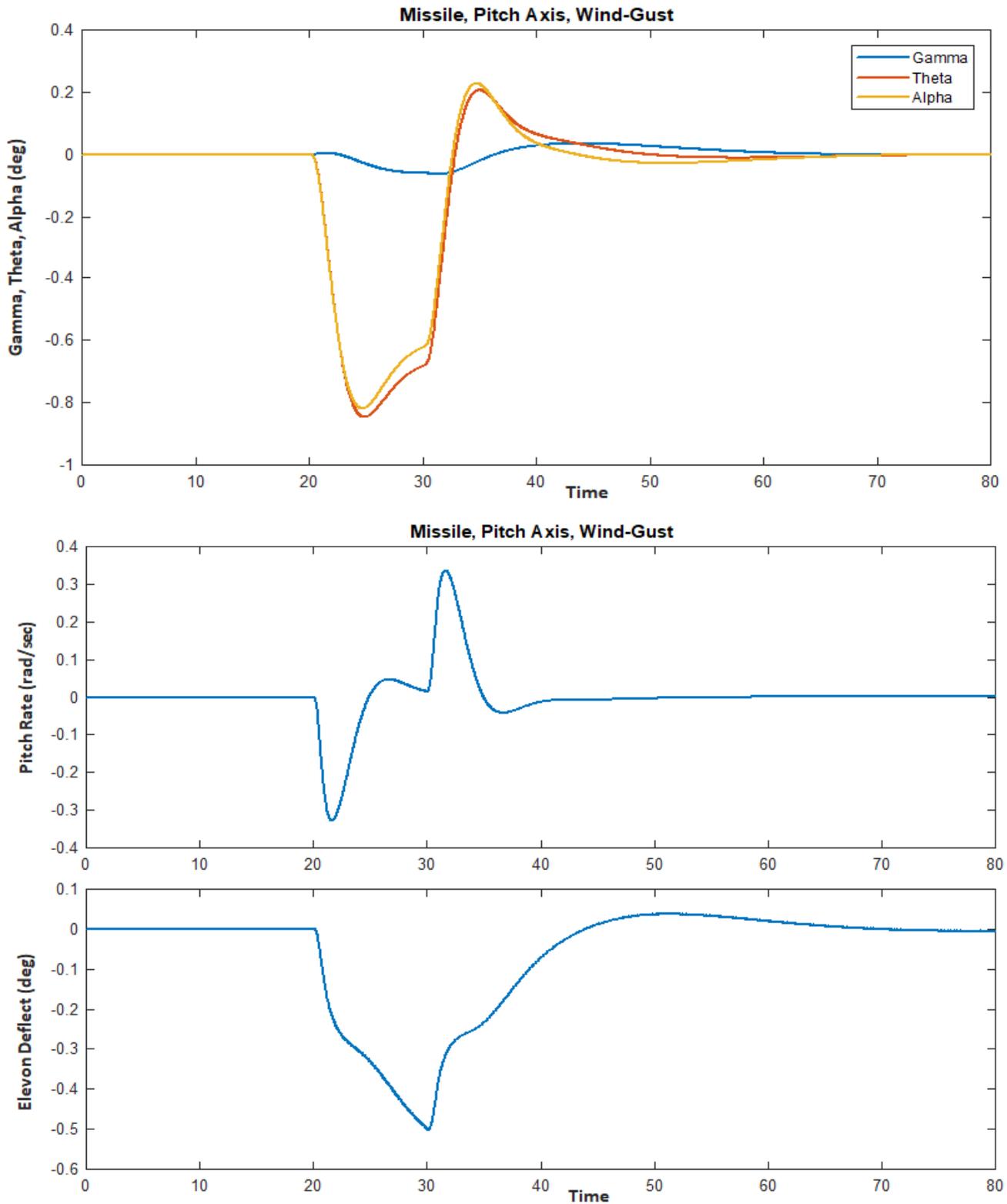


Figure 2.4 The Missile is excited by a Wind-Gust from below that causes Negative pitch and rate and Z-acceleration. The Elevon Responds with Negative Deflection to Counteract the Negative Pitch Rate

2.3 Stability Analysis

The system stability is analyzed in the frequency domain by calculating the Nichols plot using the open-loop Simulink model "Open_Pitch.mdl" shown in Figure 2.5. The script file "frequ.m" calculates the frequency response across the opened loop. The model includes the first order actuator and a low-pass filter. The loop is broken between the low-pass filter output and the actuator input. Figure 2.6 shows the control system's stability margin.

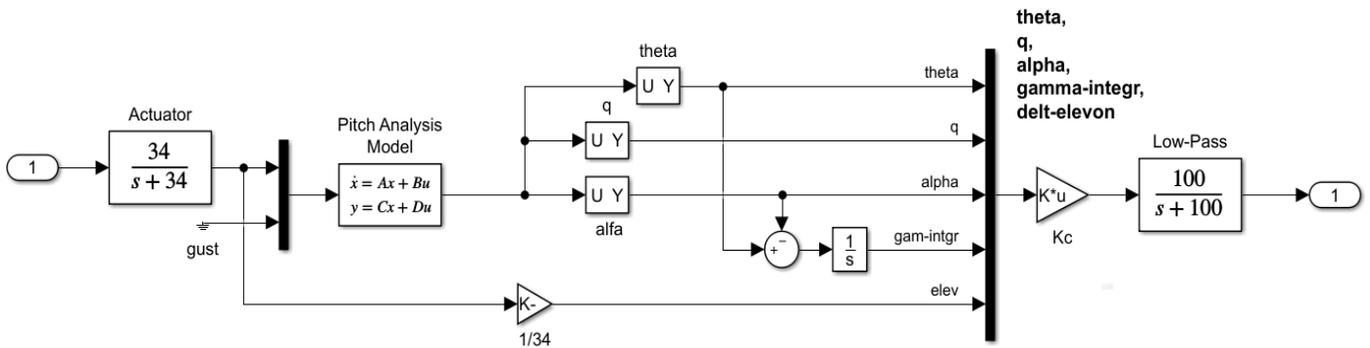


Figure 2.5 Open-Loop Model "Open_Pitch.mdl" for Pitch Stability Analysis

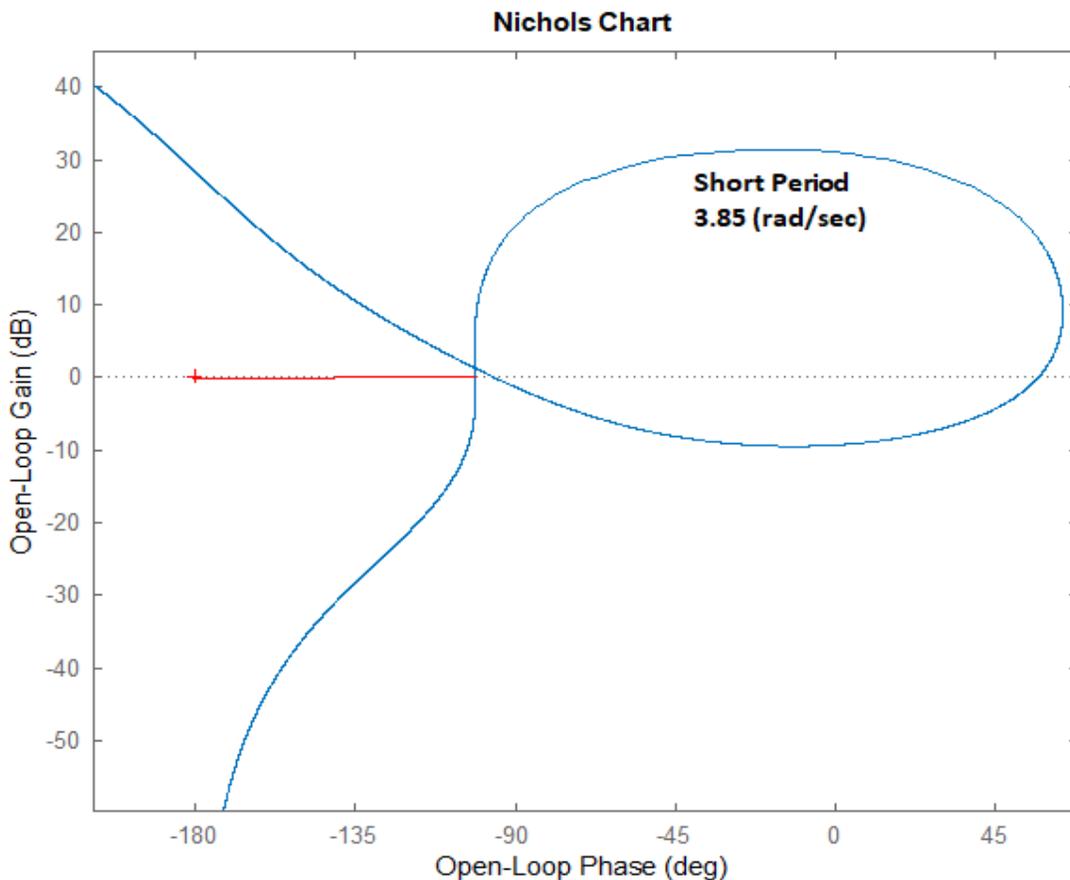


Figure 2.6 The Nichols Plot Shows that the LQR Control System has Plenty of Stability Margin in Pitch. The System also has a Short-Period Mode at 3.85 (rad/sec)

3.1 Lateral Control Design and Analysis

The input file for the coupled Roll and Yaw axes design is “*Later_LQR_Des.Inp*” located in subdirectory “*Control Analysis\LQG\Examples\Missile Control Design\Lateral LQR*”. It contains several Flixan datasets that generate plant models and perform steady-state LQR state-feedback control design. They are processed by a batch set located at the top of the file. The batch first retains the control weight matrices Q_c and R_c from getting erased in systems file “*Later_LQR_Des.Qdr*”. Then it generates the vehicle model “*Missile with Wing, Mach: 2.5, Qbar: 1220*” that includes both pitch and lateral dynamics. The initial lateral design model is then extracted from the above system and saved as “*Missile with Wing Lateral Design Model*”. It consists of two inputs, Aileron and Rudder deflections in (rad), and 5 outputs: roll attitude and rate, yaw attitude and rate and the angle of sideslip in radians. A second longitudinal system is also created with title: “*Missile with Wing Lateral Analysis Model*”. It includes a wind-gust velocity input in (feet/sec) and other outputs, and it will be used in simulations. The direction of the gust is different than the pitch model. It is now perpendicular to the vehicle x-axis, and along the $-y$ direction to excite the roll and yaw dynamics, as defined in the vehicle input data by the wind azimuth and elevation angles (90° and 90°).

```
BATCH MODE INSTRUCTIONS .....
Batch for Designing Lateral Models and Gains for a Missile with Wing
!
! This batch set creates the Design and Analysis models for a
! Missile with Wing at 2.5 Mach, and performs LQR design.
! The Missile has a fixed Thrust and it is controlled by 3 Aerosurfaces
!
!           Control Design Matrices
Retain Matrix   : State Weight Matrix Qc (9x9)
Retain Matrix   : Control Weight Matrix Rc (2x2)
!
Flight Vehicle  : Missile with Wing, Mach: 2.5, Qbar: 1220
System Modificat : Missile with Wing Lateral Design Model
System Modificat : Missile with Wing Lateral Analysis Model
Transf-Functions : Actuator: 34/(s+34)
Transf-Functions : Integrator
System Connection: Augmented Lateral Design Model
System Modificat : Augmented Lateral Design Model-2
LQR Control Des  : LQR Control Design for Augmented Lateral Design Model
!
!           Convert to Matlab
To Matlab Format : Missile with Wing Lateral Analysis Model
To Matlab Format : LQR State-Feedback Control for Augmented Lateral Design Model
```

FLIGHT VEHICLE INPUT DATA

Missile with Wing, Mach: 2.5, Qbar: 1220

**! Rigid-Body Missile controlled by 3 aerosurfaces. The engine has fixed thrust
! and does not gimbal**

Body Axes Output, Attitude=Euler Angles, NoWind Alpha

Vehicle Mass (lb-sec²/ft), Gravity Accelerat. (g) (ft/sec²), Earth Radius (Re) (ft) : 1219.1, 32.07,
Moments and products of Inertias Ixx, Iyy, Izz, Ixy, Ixz, Iyz, in (lb-sec²-ft) : 0.4063E+04 0.1654E+06
CG location with respect to the Vehicle Reference Point, Xcg, Ycg, Zcg, in (feet) : 26.19, 0.0, -0.15
Vehicle Mach Number, Velocity Vo (ft/sec), Dynamic Pressure (psf), Altitude (feet) : 2.5, 2427.4,1220.6,
Inertial Acceleration Vo_dot, Sensed Body Axes Accelerations Ax,Ay,Az (ft/sec²) : 60.0, 60.0, 0.0, 10.5
Angles of Attack and Sideslip (deg), alpha, beta rates (deg/sec) : 10.5, 0.0, 0.0, 0.0
Vehicle Attitude Euler Angles, Phi_o,Thet_o,Psi_o (deg), Body Rates Po,Qo,Ro (deg/sec) : 0.0,39.6,0.0, 0.0, 0.132,
Wind Gust Vel wrt Vehi (Azim & Elev) angles (deg), or Force(lb), Torque(ft-lb), locat:xyz: Gust 90.0 90.0
Surface Reference Area (feet²), Mean Aerodynamic Chord (ft), Wing Span in (feet) : 145.4, 22.0, 22.0
Aero Moment Reference Center (Xmrc,Ymrc,Zmrc) Location in (ft), {Partial_rho/ Partial_H} : 26.19, 0.0, -0.238, 0.0
Aero Force Coef/Deriv (1/deg), Along -X, {Cao,Ca_alf,PCa/PV,PCa/Ph,Ca_alfdot,Ca_q,Ca_bet} : 0.1, 0.002, 0.0, 0.0,
Aero Force Coeff/Derivat (1/deg), Along Y, {Cyo,Cy_bet,Cy_r,Cy_alf,Cy_p,Cy_betdot,Cy_V} : 0.0, -0.023, 0.0, 0.0,
Aero Force Coeff/Deriv (1/deg), Along Z, {Czo,Cz_alf,Cz_q,Cz_bet,PCz/Ph,Cz_alfdot,PCz/PV} : -0.1, -0.032, 0.0, 0.0,
Aero Moment Coeff/Derivat (1/deg), Roll: {Clo, Cl_beta, Cl_betdot, Cl_p, Cl_r, Cl_alfa} : 0.0, -0.0017,0.0,-0.243,
Aero Moment Coeff/Deriv (1/deg), Pitch: {Cmo,Cm_alfa,Cm_alfdot,Cm_bet,Cm_q,PCm/PV,PCm/Ph} : -0.037,-0.011, 0.0,0.0,
Aero Moment Coeff/Derivat (1/deg), Yaw : {Cno, Cn_beta, Cn_betdot, Cn_p, Cn_r, Cn_alfa} : 0.0, 5.6e-4, 0.0,0.1388,

Number of Control Surfaces, With or No TWD (Tail-Wags-Dog and Hinge Moment Dynamics) ? : 3 No TWD

Control Surface No: 1 Elevator
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.000
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00003 -0.0 -0.00087, 0.00
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : 0.0 -0.0072 0.0 0.0

Control Surface No: 2 Aileron
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.0
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00003 0.0011 0.0 0.0
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : -6.54e-4 0.0 -0.0014 0.0

Control Surface No: 3 Rudder
Trim Angle, Max/Min Deflection Angles from Trim, Hinge Line Angles: phi_h, lamda_h (deg): 0.0 30.0 -30.0 0.000
Surface Mass, Inertia about Hinge, Moment Arm (Hinge to Surface CG), Surface Chord, Area : 0.0 0.0 0.0 0.0
Hinge Moment Derivatives (1/deg), { Chm_Alpha, Chm_Beta, Chm_Delta, Chm_Mach } : 0.0 0.0 0.0 0.0
Location of the Hinge Line Center with respect to Vehicle Reference (feet), {Xcs,Ycs,Zcs}: 0.0 0.0 0.0
Forces (-x,y,z) due to Deflect. and Rates {Ca_del,Cy_del,Cz_del, Ca_deld,Cy_deld,Cz_deld} : 0.00001 0.0034 0.0 0.0
Moments due to Deflections and Rates {Cl_del,Cm_del,Cn_del,Cl_deldot,Cm_deldot,Cn_deldot} : 5.9456e-4 0.0 -0.0035

Number of Bending Modes : 0

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Missile with Wing Lateral Design Model

Missile with Wing, Mach: 2.5, Qbar: 1220

! The initial Lateral Design system is extracted from the coupled RB system above

!
TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract Inputs : 2 3
Extract States : 1 2 5 6 8
Extract Outputs: 1 2 5 6 8

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Missile with Wing Lateral Analysis Model

Missile with Wing, Mach: 2.5, Qbar: 1220

! The lateral Analysis/ Simulation system is extracted from the coupled RB system above

!
TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract Inputs : 2 3 4
Extract States : 1 2 5 6 8
Extract Outputs: 1 2 5 6 8 11 13

The system modification datasets extract the lateral variables from the coupled system "Missile with Wing, Mach: 2.5, Qbar: 1220" and save them in file "Later_LQR_Des.Qdr" as separate systems.

In the lateral direction we would like to command and track the heading direction angle (ξ). The heading angle can be controlled by a coordinated roll and yaw command that can be achieved with good roll and yaw attitude tracking performance. We introduce therefore two additional states in the design model: ϕ -integral and ψ -integral because we want to be able to command them independently in order to minimize the β -transients. It is also good idea to include simple aileron and rudder actuator models in the synthesis model because it introduces more plant information in the design and makes the control system more efficient with less phase-lag. We introduce therefore two additional states in the state-vector: δ_{aileron} and δ_{rudder} , a total of 9 states. This will create a (2×9) state-feedback LQR gain matrix.

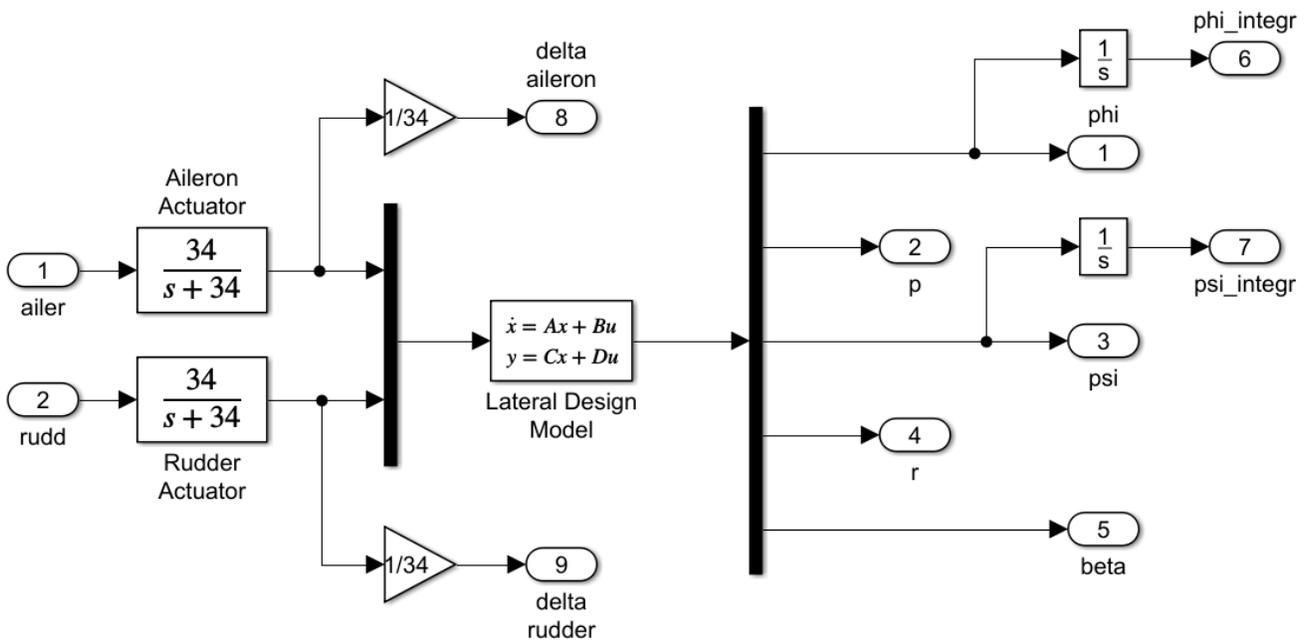


Figure 3.1 Augmented Lateral Design Plant for LQR Control Design

Figure 3.1 shows the augmented plant for the Roll/ Yaw LQR control design. The following interconnection dataset combines the 5 subsystems together and generates the augmented system, which is: “*Augmented Lateral Design Model*”. The sequence of the states, however, is not the same as the outputs sequence and it is modified by reordering the states to “*Augmented Lateral Design Model-2*” which conveniently makes the C matrix equal to the identity I_9 .

INTERCONNECTION OF SYSTEMS

Augmented Lateral Design Model

! Create a 9-State Augmented Model that Includes Phi-integr, Psi-integr,
! Aileron and Rudder deflections in the state vector for Lateral Control Design
!

Titles of Systems to be Combined

Title 1 Actuator: $34/(s+34)$

Title 2 Actuator: $34/(s+34)$

Title 3 Missile with Wing Lateral Design Model

Title 4 Integrator

Title 5 Integrator

SYSTEM INPUTS TO SUBSYSTEM 1

System Input 1 to Subsystem 1, Input 1, Gain= 1.0

to Aileron Actuator
Delta-ailer Command

SYSTEM INPUTS TO SUBSYSTEM 2

System Input 2 to Subsystem 2, Input 1, Gain= 1.0

to Rudder Actuator
Delta-ruddr Command

SYSTEM OUTPUTS FROM SUBSYSTEM 3

System Output 1 from Subsystem 3, Output 1, Gain= 1.0

System Output 2 from Subsystem 3, Output 2, Gain= 1.0

System Output 3 from Subsystem 3, Output 3, Gain= 1.0

System Output 4 from Subsystem 3, Output 4, Gain= 1.0

System Output 5 from Subsystem 3, Output 5, Gain= 1.0

Vehicle Plant
Phi
p - roll rate
Psi
r - yaw rate
Beta

SYSTEM OUTPUTS FROM SUBSYSTEM 4

System Output 6 from Subsystem 4, Output 1, Gain= 1.0

Integrator
Phi-integral

SYSTEM OUTPUTS FROM SUBSYSTEM 5

System Output 7 from Subsystem 5, Output 1, Gain= 1.0

Integrator
Psi-integral

SYSTEM OUTPUTS FROM SUBSYSTEM 1

System Output 8 from Subsystem 1, Output 1, Gain= 0.0294118

Actuator
delta-aileron

SYSTEM OUTPUTS FROM SUBSYSTEM 2

System Output 9 from Subsystem 2, Output 1, Gain= 0.0294118

Actuator
delta-rudder

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 3

Subsystem 1, Output 1 to Subsystem 3, Input 1, Gain= 1.0000

Ailer-Actuat to Vehicle
Aileron-deflect

SUBSYSTEM NO 2 GOES TO SUBSYSTEM NO 3

Subsystem 2, Output 1 to Subsystem 3, Input 2, Gain= 1.0000

Ruddr-Actuat to Vehicle
Rudder-deflect

SUBSYSTEM NO 3 GOES TO SUBSYSTEM NO 4

Subsystem 3, Output 1 to Subsystem 4, Input 1, Gain= 1.0000

Vehicle to Integrator-4
Phi

SUBSYSTEM NO 3 GOES TO SUBSYSTEM NO 5

Subsystem 3, Output 3 to Subsystem 5, Input 1, Gain= 1.0000

Vehicle to Integrator-4
Psi

Definitions of Inputs = 2

Aileron Deflection Command (delta) rad

Rudder Deflection Command (delta) rad

Definitions of Outputs = 9

Roll Attitude, Phi (rad)

Roll Rate, p (rad/sec)

Yaw Attitude, Psi (rad)

Yaw Rate, r (rad/sec)

Angle of Sideslip beta (rad)

Phi-Integral (rad-sec)

Psi-Integral (rad-sec)

Aileron Deflection, delta-ailer (rad)

Rudder Deflection, delta-rudder (rad)

SYSTEM OF TRANSFER FUNCTIONS ...

Actuator: 34/(s+34)

! First order Actuator 34 (rad/sec) Bandwidth

Continuous

TF. Block # 1 34/(s+34)

Order of Numer, Denom= 0 1

Numer 0.0 34.0

Denom 1.0 34.0

Block #, from Input #, Gain

1 1 1.00000

.....
Outpt #, from Block #, Gain

1 1 1.00000

.....
Definitions of Inputs = 1

Delta Command

Definitions of Outputs = 1

Delta Out

SYSTEM OF TRANSFER FUNCTIONS ...

Integrator

Continuous

TF. Block # 1 (1/s)

Order of Numer, Denom= 0 1

Numer 0.0 1.0

Denom 1.0 0.0

Block #, from Input #, Gain

1 1 1.00000

.....
Outpt #, from Block #, Gain

1 1 1.00000

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Augmented Lateral Design Model-2

Augmented Lateral Design Model

! Rearrange the Order of States to be the same as the Outputs

! Makes C=Identity

TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract States : 3 4 5 6 7 8 9 1 2

Extract Outputs: 1 2 3 4 5 6 7 8 9

The dataset “*LQR Control Design for Augmented Lateral Design Model*” performs the LQR control design using the plant “*Augmented Lateral Design Model-2*”. It uses the C matrix for criteria which is Identity and the (9x9) weight matrix Q_c penalizes the individual states. The (2x2) matrix R_c penalizes the two controls, which are: aileron and rudder activity. The weight matrices are already set in the systems file. The LQR program generates the (2x9) state-feedback matrix K_{pr} that stabilizes the plant by closing the control loop between the state-vector and the two aerosurface inputs. The gain matrix is also saved in the systems file under the title “*LQR State-Feedback Control for Augmented Lateral Design Model*”. The matrix K_{pr} and the lateral analysis model are also saved in Matlab format as “*Kpr.mat*” and “*vehi_lateral.m*” respectively for further analysis.

```

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design for Augmented Lateral Design Model
Plant Model Used to Design the Control System: Augmented Lateral Design Model-2
Criteria Optimization Output is Matrix C
State Penalty Weight (Qc) is Matrix: Qc9 State Weight Matrix Qc (9x9)
Control Penalty Weight (Rc) is Matrix: Rc2 Control Weight Matrix Rc (2x2)
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kpr LQR State-Feedback Control for Augmented Lateral Design Model
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Control for Augmented Lateral Design Model
Matrix Kpr
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Missile with Wing Lateral Analysis Model
System
Vehi_lateral.m
-----
END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END-END

```

3.2 Lateral Simulation

The closed-loop simulation model “*Lateral_Sim.mdl*” in Figure 3.2 is located in folder “*Control Analysis\LQG\ Examples\Missile Control Design\Lateral LQR*” and it is used to analyze the system’s response to gusts and to heading commands. The 9-state-feedback loop is closed via matrix K_{pr} which includes: ϕ -integral, ψ -integral, $\delta_{aileron}$ and δ_{rudder} feedback in addition to the feedback from the original vehicle states: $(\phi, p, \psi, r, \beta)$. The heading direction is $\xi = \psi + \beta$. The heading error is converted into a simultaneously applied roll and yaw attitude command which makes the vehicle to perform a coordinated roll/ yaw turn with minimal β -transient.

In Figure 3.3 the vehicle is commanded to perform a 10° increment in ξ which is achieved by a coordinated roll and yaw command to minimize the β -transients which are undesirable at high Q_{bar} . In Figure 3.4 the missile is excited by a lateral wind-gust velocity impulse along the $-y$ direction and it responds by rotating the aerosurfaces.

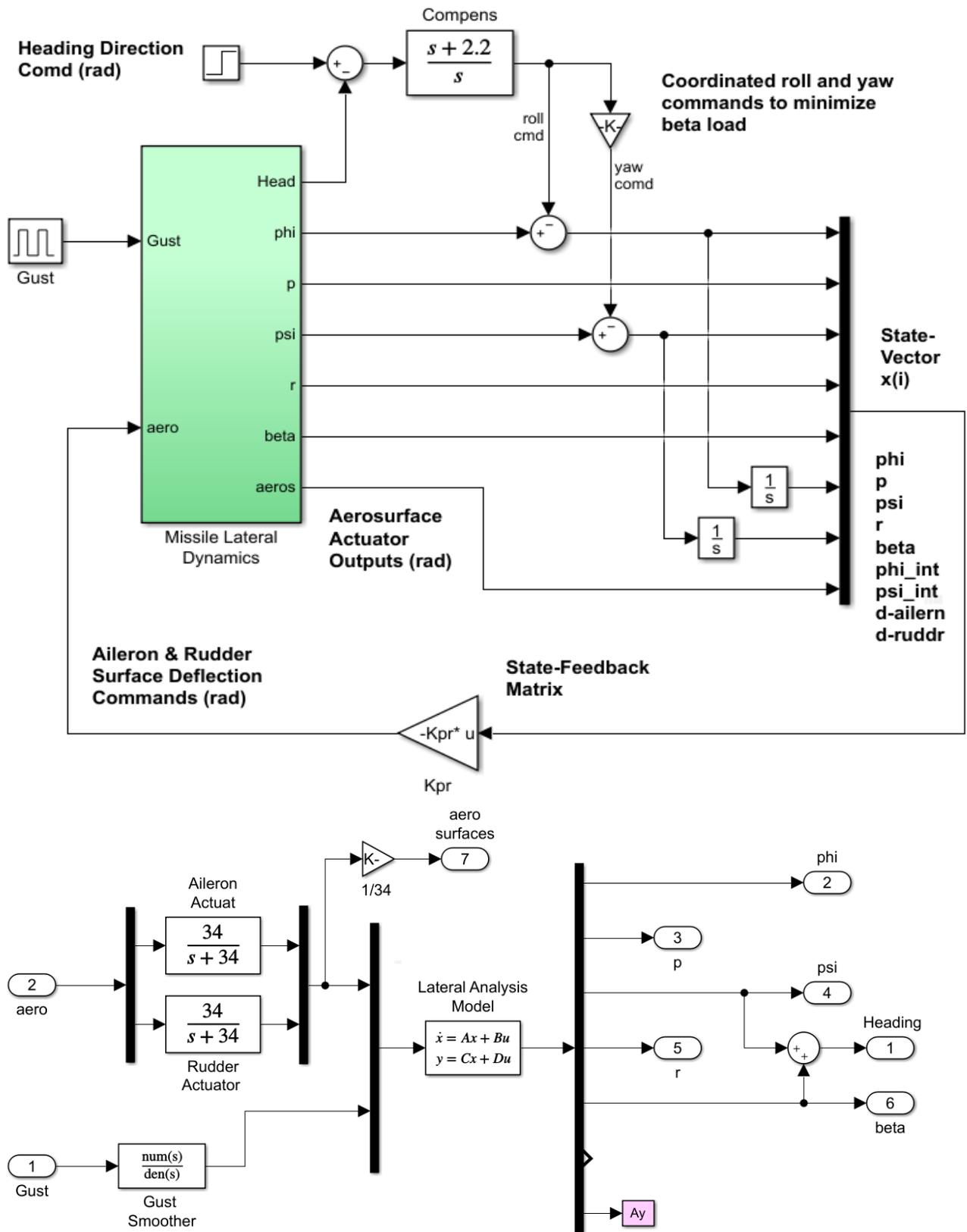


Figure 3.2 Roll and Yaw Axes Closed-Loop Simulation Model "Lateral_Sim.mdl"

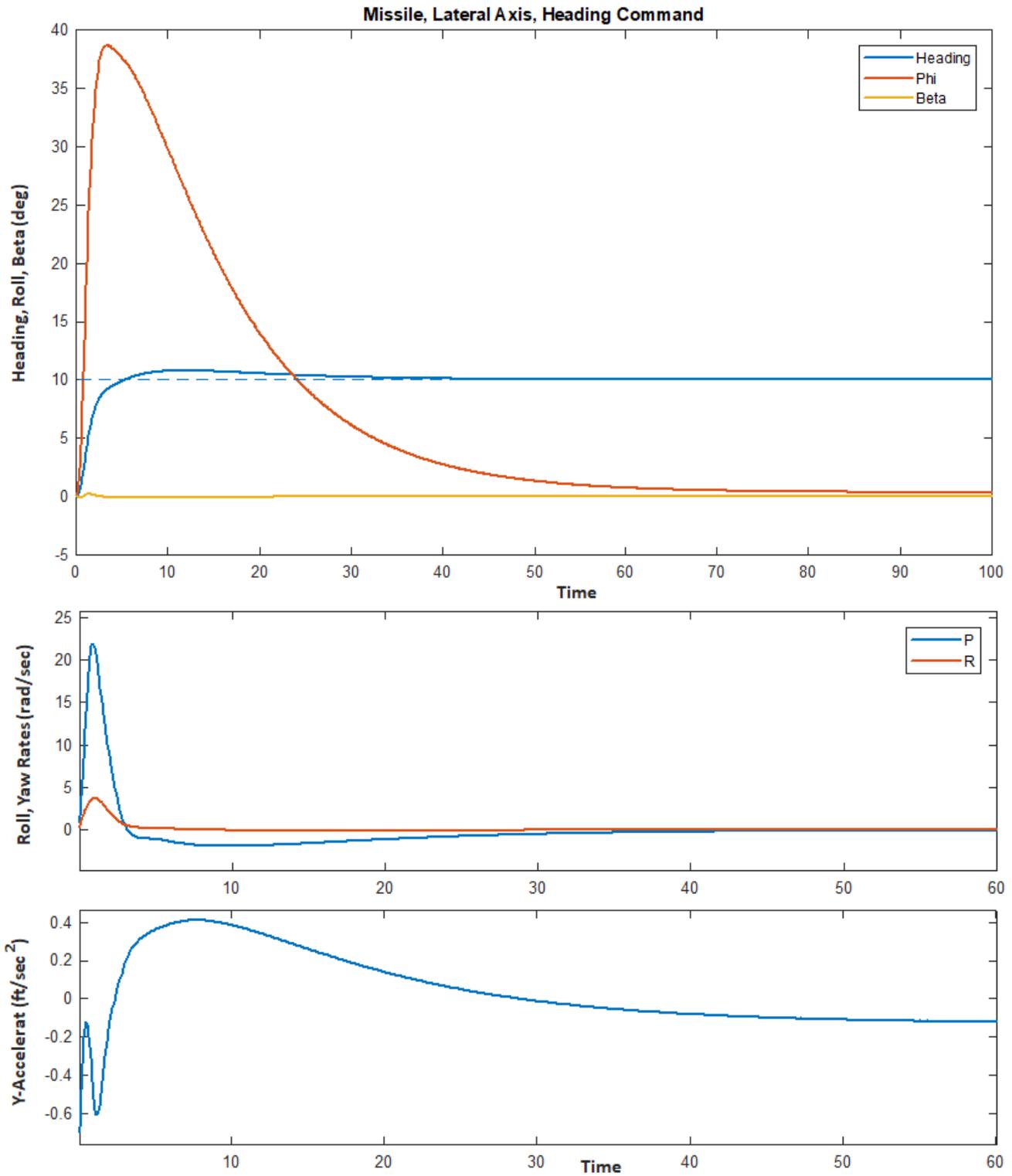


Figure 3.3 Missile Responds to 10° Heading Command. Performs Coordinated Roll/ Yaw Maneuver (mostly roll) to Change its Heading. Beta Transient is minimized by the Coordinated Roll/ Yaw Turn

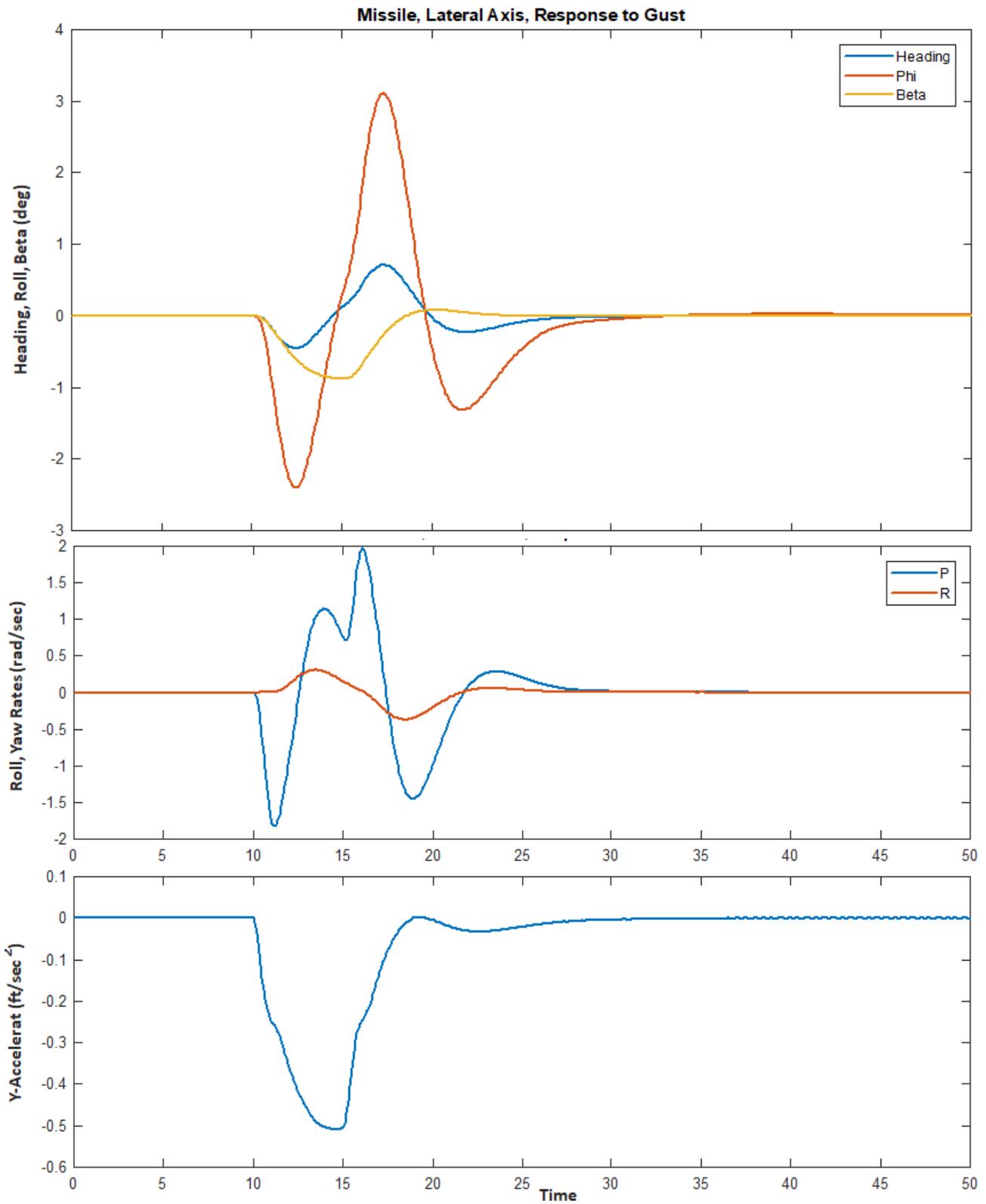


Figure 3.4 Missile is excited by Lateral Wind-Gust from the right side, along $-Y$, causing $-Y$ Acceleration. It also causes Negative Roll and Positive Yaw due to the Vertical Stabilizer. Beta is Initially Negative because it does not see the Wind due to (NoWind Alpha/ Beta) Definition in the Data. The aileron and Rudder accordingly respond to counteract the Vehicle Roll and Yaw Rates

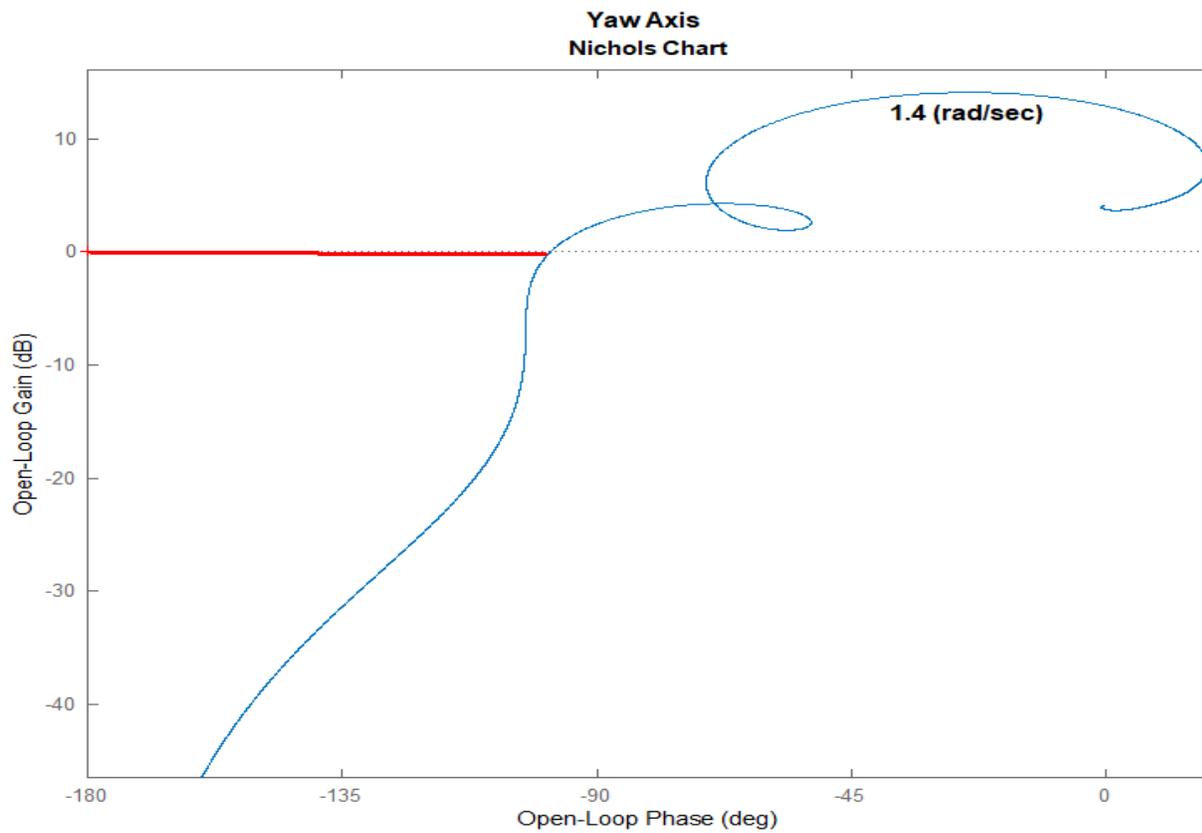
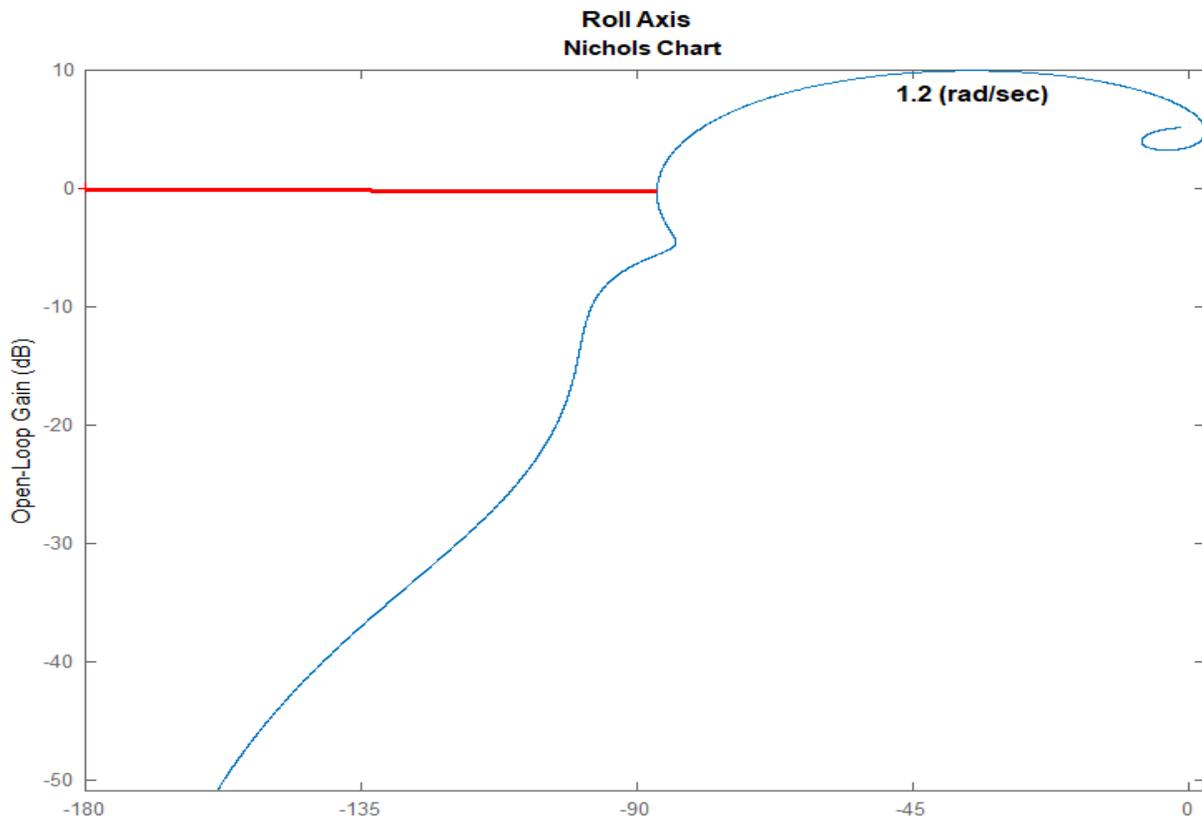
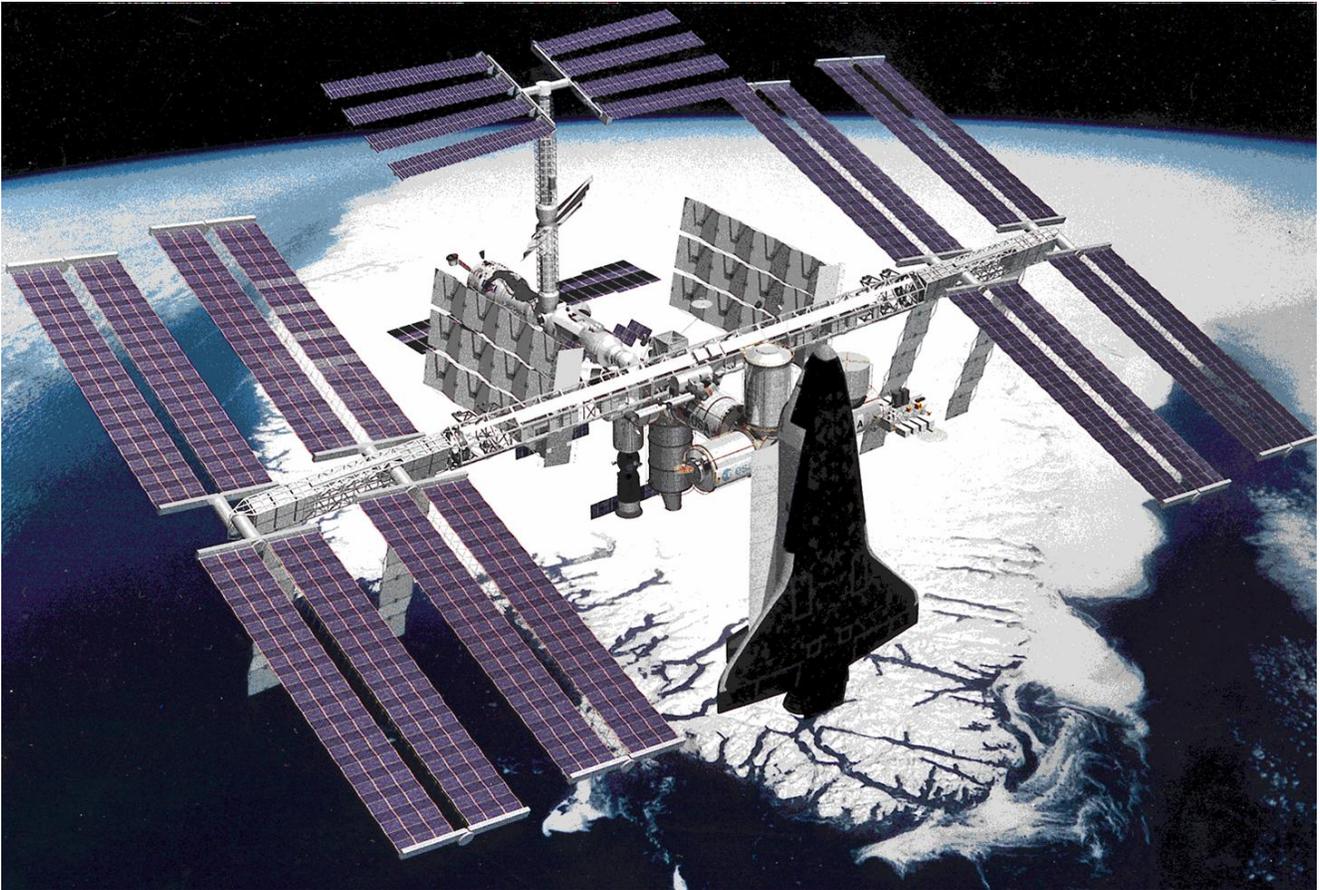


Figure 3.6 Nichols Plots Showing Stability of the LQR Control System in the Roll and Yaw Axes. The System also has a Short-Period Modes at 1.2 and 1.4 (rad/sec)

Space Station Attitude and Momentum Control Design



In this example we design a control system for a large and flexible Space Station that is in orbit around the earth. The Space Station consists of a truss structure with some attached modules for the crew, equipment, experiments, etc. which are located near the center of the structure. The attitude control system uses reaction control jets (RCS) and control moment gyros (CMG) but in this example we will examine only CMG control and design a system that stabilizes the Station attitude and manages the CMG momentum from saturating. The Space Station orbit is circular and its attitude is almost constant relative to the Local Vertical Local Horizontal (LVLH) frame. The LVLH x-axis is in the direction of the velocity vector, the z-axis is pointing towards the earth center and the y-axis towards the right solar array.

The Attitude Control System (ACS) has different modes of operation and one of the control modes is the TEA seeking mode where the spacecraft attitude converges to one of the Torque Equilibrium Attitudes (TEA). This happens when the ACS balances the average aerodynamic torques with the gravity gradient torques. The CMGs are considered as a cluster located near the center and not modeled individually. They are momentum exchange devices that have limited torque and momentum. Momentum is the integral of torque, that is, they can only supply torque for a limited time before they saturate and when they do they require momentum desaturation. Momentum dump is achieved either by firing RCS jets or by applying gravity gradient torques. The ACS manages the CMG momentum by positioning the spacecraft attitude at the TEA, and therefore, without firing the RCS jets. When operating in this mode the Station converges to the average TEA and avoids secular CMG momentum

build up. When the momentum begins to grow in a certain direction due to external disturbances, the ACS changes the attitude in the direction that reduces momentum.

The Station has a horizontal boom with two rotating solar arrays which are always pointing towards the sun, completing, therefore, one rotation per orbit relative to the spacecraft. The aerodynamic disturbances cause the CMG momentum to cycle but the control system prevents it from reaching saturation. The aerodynamic disturbances consist of steady torques and also cyclic components that excite the spacecraft attitude to oscillations. There are two frequency components associated with the cyclic disturbance torques: one is at orbital rate (ω_o) due to the difference in atmospheric density between the sunny and the dark sides of the earth, and the second component is at twice the orbital rate ($2\omega_o$) caused by drag variation due to the solar arrays rotation. In this mode of operation the function of the CMG control system is not to maneuver the Space Station attitude but to stabilize it at the TEA and to attenuate the attitude oscillations which are caused by cyclic aerodynamic disturbances.

The purpose of this example is to familiarize the analyst with the Flixan program, create rigid and flexible spacecraft models and use them for control design. We will develop linear and non-linear rigid and flexible spacecraft models, design LQR state-feedback for attitude control and momentum management. Since the spacecraft is stabilized in the LVLH frame the model attitude and rates is calculated in the LVLH frame. We will also analyze the system's stability and performance by using simulations and frequency response analysis.

1. Dynamic Model

The linear dynamic model in Equation 1 calculates the spacecraft attitude, rate and CMG momentum relative to the LVLH frame. The linearized gravity gradient torque is also included in the model. It is a function of attitude (ϕ, θ, ψ) and therefore it can be used to regulate the CMG momentum by adjusting the station attitude. The spacecraft states are: LVLH attitude, LVLH rates, and CMG momentum in body axes. The CMG momentum integral is also included to help bound the CMG momentum. The inputs are the CMG control torques T_c and external disturbances T_d . It also includes the linearized gravity gradient dynamics which are functions of the LVLH Euler angles.

$$\begin{aligned}
& \begin{bmatrix} I_{XX} & I_{XY} & I_{XZ} \\ I_{XY} & I_{YY} & I_{YZ} \\ I_{XZ} & I_{YZ} & I_{ZZ} \end{bmatrix} \begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \omega_o \begin{bmatrix} 0 & 2I_{YZ} & I_{XX} - I_{YY} + I_{ZZ} \\ -2I_{YZ} & 0 & 2I_{XY} \\ -I_{XX} + I_{YY} - I_{ZZ} & -2I_{XY} & 0 \end{bmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \\
& + \omega_o^2 \begin{bmatrix} 4(I_{ZZ} - I_{YY}) & 3I_{XY} & -I_{XZ} \\ 4I_{XY} & 3(I_{ZZ} - I_{XX}) & I_{YZ} \\ -4I_{XZ} & -3I_{YZ} & (I_{XX} - I_{YY}) \end{bmatrix} \begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} + \omega_o^2 \begin{pmatrix} -4I_{YZ} \\ 3I_{XZ} \\ I_{XY} \end{pmatrix} + T_d + T_c \\
& \begin{pmatrix} \dot{h}_X \\ \dot{h}_Y \\ \dot{h}_Z \end{pmatrix}_{CMG} = \omega_o \begin{pmatrix} h_Z \\ 0 \\ -h_X \end{pmatrix}_{CMG} - T_c
\end{aligned}$$

Equation 1 Linearized Spacecraft Dynamic Equations with Attitude in the LVLH frame

2. Design Concepts

At the torque equilibrium attitude (TEA) the CMG momentum is cyclic but it does not diverge to saturation. We can, therefore, design a control system that converges the Space Station attitude to the TEA, which is a stable position because the average aerodynamic torque balances with the average gravity gradient torque. This is accomplished by applying feedback from the CMG momentum to prevent it from diverging and by this process the system attitude converges to the TEA. The LQR control method will be used to design the state-feedback gain that stabilizes the coupled dynamic system. It optimizes a performance index that is defined by the state and control weight matrices Q_c and R_c . The LQR method requires a linear dynamic model to synthesize the controller and the Flixan design model described in Equation 1 will be used as a synthesis model. The state feedback control law, therefore, stabilizes not only the attitude but also the CMG momentum by keeping it oscillating around zero. It prevents it from diverging to saturation and as a result the spacecraft attitude converges to the TEA. The CMG momentum, however, is not unstable but it is cycling because it responds to the cyclic aerodynamic disturbances. This is a continuous momentum desaturation method which is very attractive because it does not require RCS propellant. It adjusts the spacecraft attitude and uses gravity gradient to prevent the CMG momentum from building up. It relies on sufficient knowledge of the vehicle mass properties for the derivation of the control gains.

In the following sections will present two designs: a simple state-feedback design, and a more complex design that includes disturbance attenuation filters. We will analyze both designs and compare results.

3. Simple Design without Filters

We begin with a preliminary state-feedback design that is based on a dynamic model derived from Equation 1. We will use Flixan to create the rigid and flexible spacecraft models, design the control system using LQR, and analyze stability.

3.1 Flixan Models

The files for this analysis are in directory: “*Flixan\Control Analysis\LQG\Examples\Space-Station w CMG2\Design-1*”. The Space Station parameters are defined in the input file “*Space_Station.Inp*” which includes two flight vehicle datasets: a rigid-body “*Space Station with Double-Gimbal CMG Array (Rigid)*”, and a flexible model with 34 structural modes “*Space Station with Double-Gimbal CMG Array (Flex)*”. The modes are already selected and scaled and they are located in dataset “*Space Station with Double-Gimbal CMG Array, 34 Flex Modes*”. The “**LVLH Attitude & Rate**” flag is included in the flags line that defines the output attitude and rate relative to the LVLH frame. It uses the -0.063 (rad/sec) pitch rate which is equal to the orbital rate ω_0 for the transformation. A batch set is included at the top of the input file with title “*Batch for Large Flexible Space Station*”. It is shown below and it is used for processing the entire input file fast.

```
BATCH MODE INSTRUCTIONS .....
Batch for Large Flexible Space Station
! This batch set creates a model for a Space Station that is
! controlled by an array of double-gimbal CMGs. Two models
! are created, a rigid-body model and a flexible model using the
! attached modal data. The design model is extracted from Rigid Vehicle
! and the plant state is transformed so that it is equal to the Output, C=I
!
Retain Matrix      : State Weight Matrix Qc (12x12)
Retain Matrix      : Control Weight Matrix Rc (3x3)
!
Flight Vehicle     : Space Station with Double-Gimbal CMG Array (Rigid)
Flight Vehicle     : Space Station with Double-Gimbal CMG Array (Flex)
System Modificat   : Space Station with Double-Gimbal CMG Array (Design Plant)
System Modificat   : Space Station with Double-Gimbal CMG Array (LVLH Plant)
Transf-Function    : 3 Integrators
System Connection  : Augmented Design Plant
LQR Control Des    : LQR Control Design for Augmented Space Station Model
!
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Rigid)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Flex)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Design Plant)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (LVLH Plant)
To Matlab Format    : Augmented Design Plant
To Matlab Format    : LQR State-Feedback Gain for Augmented Design Plant
-----
```

FLIGHT VEHICLE INPUT DATA

Space Station with Double-Gimbal CMG Array (Rigid)

!
! The Space Station state-space model is created using the vehicle modeling program.
! The model uses an array of double-gimbal control moment gyros, 3 rate
! gyros, 3 attitude sensors, and 4 accelerometers. The Station is initialized at the
! Local Vertical Local Horizontal (LVLH) attitude and it has a negative pitch rate
! -0.063 radians/sec which is equal to the orbital rate. The vehicle rates are with
! respect to the LVLH frame. A constant bias torque 7.40153 (ft-lb) is applied
! in the direction (-0.1969, 0.5963, 0.7781) to represent the steady-state gyroscopic
! and gravity-gradient torques due to the constant pitch rate
!

Body Axes Output, LVLH Attitude & Rate

Vehicle Mass (lb-sec²/ft), Gravity Accelerat. (g) (ft/sec²), Earth Radius (Re) (ft) : 6200.0 32.174
Moments and products of Inertias Ixx, Iyy, Izz, Ixy, Ixz, Iyz, in (lb-sec²-ft) : 0.1194e+9 0.404e+8
CG location with respect to the Vehicle Reference Point, Xcg, Ycg, Zcg, in (feet) : 0.0 0.0
Vehicle Mach Number, Velocity Vo (ft/sec), Dynamic Pressure (psf), Altitude (feet) : 0.0 25500.0
Inertial Acceleration Vo_dot, Sensed Body Axes Accelerations Ax,Ay,Az (ft/sec²) : 0.0 0.0
Angles of Attack and Sideslip (deg), alpha, beta rates (deg/sec) : 0.0 0.0
Vehicle Attitude Euler Angles, Phi_o,Thet_o,Psi_o (deg), Body Rates Po,Qo,Ro (deg/sec) : 0.0000 0.000
W-Gust Azim & Elev angles (deg), or Torque/Force direction (x,y,z), Force Locat (x,y,z) : Torque 0.1969 -0.5963
Surface Reference Area (feet²), Mean Aerodynamic Chord (ft), Wing Span in (feet) : 0.0 1.0 1.0
Aero Moment Reference Center (Xmrc,Ymrc,Zmrc) Location in (ft), {Partial_rho/ Partial_H} : 0.0 0.0
Aero Force Coef/Deriv (1/deg), Along -X, {Cao,Ca_alf,PCa/PV,PCa/Ph,Ca_alfdot,Ca_q,Ca_bet} : 0.0 0.0
Aero Force Coeff/Derivat (1/deg), Along Y, {Cyo,Cy_bet,Cy_r,Cy_alf,Cy_p,Cy_betdot,Cy_V} : 0.0 -0.0
Aero Force Coeff/Deriv (1/deg), Along Z, {Czo,Cz_alf,Cz_q,Cz_bet,PCz/Ph,Cz_alfdot,PCz/PV} : 0.0 -0.0
Aero Moment Coeff/Derivat (1/deg), Roll: {Clo, Cl_beta, Cl_betdot, Cl_p, Cl_r, Cl_alfa} : 0.0 -0.0
Aero Moment Coeff/Deriv (1/deg), Pitch: {Cmo,Cm_alfa,Cm_alfdot,Cm_bet,Cm_q,PCm/PV,PCm/Ph} : 0.0 -0.0
Aero Moment Coeff/Derivat (1/deg), Yaw : {Cno, Cn_beta, Cn_betdot, Cn_p, Cn_r, Cn_alfa} : 0.0 0.0

Number of External Torques on the Vehicle : 3
Torque No 1 Direction (x, y, z) : 1.0 0.0 0.0
Torque No 2 Direction (x, y, z) : 0.0 1.0 0.0
Torque No 3 Direction (x, y, z) : 0.0 0.0 1.0

Double Gimbal Control Moment Gyro System (3-axes), Initial Momentum (x,y,z) (ft-lb-sec) : Yes 0.0 0.0 0.0

Number of Gyros, (Attitude and Rate) : 6
Gyro No 1 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Roll Rate 0.0 82.0
Gyro No 2 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Pitch Rate 0.0 82.0
Gyro No 3 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Yaw Rate 0.0 82.0
Gyro No 4 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Roll Attitu 0.0 82.0
Gyro No 5 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Pitch Attitu 0.0 82.0
Gyro No 6 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat, Node 2 : Yaw Attitu 0.0 82.0

Number of Accelerometers, Along Axes: (x,y,z) : 4
Acceleromet No 1 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 2 : X-axis Accelerat. 0.0
Acceleromet No 2 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 2 : Z-axis Accelerat. 0.0
Acceleromet No 3 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 4 : X-axis Accelerat. 0.0
Acceleromet No 4 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 4 : Z-axis Accelerat. 0.0

Number of Bending Modes : 0

The input file includes a system modification dataset “*Space Station with Double-Gimbal CMG Array (Design Plant)*” which creates the control design model by extracting a reduced number of inputs, states, and outputs from the rigid model. The systems and matrices created are saved in file “*Space_Station.Qdr*” under the same title.

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)

Space Station with Double-Gimbal CMG Array (Design Plant)

Space Station with Double-Gimbal CMG Array (Rigid)

! Create the Station Design Model by Reducing the Rigid-Body Model

! Inputs are the CMG Control Torques

! Outputs are: LVLH attitude and rates, and CMG Momentum

!

TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS

Extract Inputs : 1 2 3
Extract States : 1 3 5 2 4 6 11 12 13
Extract Outputs: 1 3 5 2 4 6 16 17 18

An additional transformation is needed because although the design plant attitude and rates are in the LVLH frame, some of the states, however, are still in the body frame. Since we assume that the state-feedback measurements are in the LVLH we want our design model state to be also in the LVLH frame. This transformation makes the design plant states to be equal to the outputs which are already LVLH attitudes and rates, and the output matrix $C=Identity$. The dataset below performs this transformation. Its title is “*Space Station with Double-Gimbal CMG Array (LVLH Plant)*”. Notice that the transformation requires the design plant matrix C to be square and observable.

```

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)
Space Station with Double-Gimbal CMG Array (LVLH Plant)
Space Station with Double-Gimbal CMG Array (Design Plant)
! Transform the design plant and make the States equal to the Outputs
SYSTEM TRANSFORMATION, STATES EQUAL TO OUTPUTS
-----

SYSTEM OF TRANSFER FUNCTIONS ...
3 Integrators
! Used to calculate the CMG Momentum Integral for the LQR Optimization
Continuous
TF. Block # 1 Integrator x                               Order of Numer, Denom= 0 1
Numer 0.0      1.0
Denom 1.0      0.0
TF. Block # 2 Integrator y                               Order of Numer, Denom= 0 1
Numer 0.0      1.0
Denom 1.0      0.0
TF. Block # 3 Integrator z                               Order of Numer, Denom= 0 1
Numer 0.0      1.0
Denom 1.0      0.0
.....
Block #, from Input #, Gain
      1      1      1.0
      2      2      1.0
      3      3      1.0
.....
Outpt #, from Block #, Gain
      1      1      1.0
      2      2      1.0
      3      3      1.0
.....
Definitions of Inputs = 3
CMG Momentum-X
CMG Momentum-Y
CMG Momentum-Z

Definitions of Outputs = 3
CMG Momentum-X Integral
CMG Momentum-Y Integral
CMG Momentum-Z Integral
-----

```

The design model is finally augmented by including 3 additional states, the (x, y, z) momentum integral because it is not sufficient to only bound the momentum from diverging but we also want it to be cycling near zero without a bias. The 3 momentum integrators are implemented as 3 transfer functions using the transfer functions combination utility. It creates the state-space system “3 Integrators”, shown above. We must also create the “*Augmented Design Plant*” by combining the LVLH design plant with the 3 additional (x, y, z) momentum integral states using the Flxan systems combination utility, see Fig.2.

INTERCONNECTION OF SYSTEMS

Augmented Design Plant

! Augment the Design Plant by including the Momentum Integral and the
! Oscillation Filters

Titles of Systems to be Combined (Found in File Comb_tst.Qdr)

Title 1 Space Station with Double-Gimbal CMG Array (LVLH Plant)

Title 2 3 Integrators

SYSTEM INPUTS TO SUBSYSTEM 1

Via Matrix +I3

3 CMG Control Torques

SYSTEM OUTPUTS FROM SUBSYSTEM 1

9 Plant States

System Output 1 from Subsystem 1, Output 1, Gain= 1.0

3 LVLH Attitudes

System Output 2 from Subsystem 1, Output 2, Gain= 1.0

System Output 3 from Subsystem 1, Output 3, Gain= 1.0

System Output 4 from Subsystem 1, Output 4, Gain= 1.0

3 LVLH Rates

System Output 5 from Subsystem 1, Output 5, Gain= 1.0

System Output 6 from Subsystem 1, Output 6, Gain= 1.0

System Output 7 from Subsystem 1, Output 7, Gain= 1.0

3 CMG Momentum

System Output 8 from Subsystem 1, Output 8, Gain= 1.0

System Output 9 from Subsystem 1, Output 9, Gain= 1.0

SYSTEM OUTPUTS FROM SUBSYSTEM 2

3 Momentum Integrals

System Output 10 from Subsystem 2, Output 1, Gain= 1.0

H-integr-X

System Output 11 from Subsystem 2, Output 2, Gain= 1.0

H-integr-Y

System Output 12 from Subsystem 2, Output 3, Gain= 1.0

H-integr-Z

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 2

to Momentum Integrat

Subsystem 1, Output 7 to Subsystem 2, Input 1, Gain= 1.0

CMG Momentm-X

Subsystem 1, Output 8 to Subsystem 2, Input 2, Gain= 1.0

CMG Momentm-Y

Subsystem 1, Output 9 to Subsystem 2, Input 3, Gain= 1.0

CMG Momentm-Z

Definitions of Inputs = 3

Roll CMG Momentum (ft-lb)

Ptch CMG Momentum (ft-lb)

Yaw CMG Momentum (ft-lb)

Definitions of Outputs = 12

Roll Attitude (phi-LVLH) (radians)

Pitch Attitude (thet-LVLH) (radians)

Yaw Attitude (psi-LVLH) (radians)

Roll Rate (p-lvlh) (rad/sec)

Pitch Rate (q-lvlh) (rad/sec)

Yaw Rate (r-lvlh) (rad/sec)

CMG Momentum in X-axis (ft-lb-sec)

CMG Momentum in Y-axis (ft-lb-sec)

CMG Momentum in Z-axis (ft-lb-sec)

CMG Momentum Integral X (ft-lb-sec^2)

CMG Momentum Integral Y (ft-lb-sec^2)

CMG Momentum Integral Z (ft-lb-sec^2)

The final dataset included in the input file is the “LQR Control Design for Augmented Space Station Model” which calculates the (3x12) state-feedback matrix K_{pqr} . The state and control weight matrices Q_c and R_c are already included in file “Space_Station.Qdr”. The state-feedback matrix K_{pqr} is also saved in file “Space_Station.Qdr” and its title is “LQR State-Feedback Gain for Augmented Design Plant”.

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN

LQR Control Design for Augmented Space Station Model

Plant Model Used to Design the Control System from:

Augmented Design Plant

Criteria Optimization Output is Matrix C

State Penalty Weight (Q_c) is Matrix: Q_{c12}

State Weight Matrix Q_c (12x12)

Control Penalty Weight (R_c) is Matrix: R_{c3}

Control Weight Matrix R_c (3x3)

Continuous LQR Solution Using Assymptotic Method

LQR State-Feedback Control Gain Matrix K_{pqr}

LQR State-Feedback Gain for Augmented Design Plant

Six Matlab conversion datasets are also included in “*Space_Station.Inp*”. They convert the systems and matrices in m-file format that can be loaded into Matlab by running the script file start.m.

```

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Rigid)
System
vehicle_rigid
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Flex)
System
vehicle_flex34
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Design Plant)
System
design_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (LVLH Plant)
System
lvlh_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Augmented Design Plant
System
augm_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Gain for Augmented Design Plant
Matrix Kpqr
-----

[Av, Bv, Cv, Dv]= vehicle_rigid;           % Load the Rigid Vehicle State-Space Model
[Af, Bf, Cf, Df]= vehicle_flex34;        % Load the Flex Vehicle State-Space Model
[Ad, Bd, Cd, Dd]= design_plant;          % Load the Rigid Design Plant Model
[Al, Bl, Cl, Dl]= lvlh_plant;            % Load the LVLH Design Plant Model
[Ag, Bg, Cg, Dg]= augm_plant;           % Load the Augmented Design Plant Model
load Kpqr -ascii;                        % Load the LQR gains from previous design

```

The files “vehicle_rigid.m” and “vehicle_flex34.m” are used in simulations. The augmented plant “augm_plant.m” can also be used for the LQR design using Matlab.

3.2 LQR Control System Design

The initialization file “start.m” loads the dynamic systems and matrices for the simulation and analysis models. In addition to Flixan, the Matlab script file “des.m” is also able to calculate the LQR control gains. It is using either the augmented design plant or by linearizing the Simulink model “Augm_Vehicle.mdl” shown in Figure 2. Figure 3 shows the rigid and flexible simulation models: “Sim_Lin_TEA_Rigid.mdl” and “Sim_Lin_TEA_Flex.mdl”, which are similar. They include the Flixan generated systems: “vehicle_rigid.m” and “vehicle_flex34.m” respectively, the second of which contains the 34 structural modes. The rate states of these systems have not been transformed to LVLH as in the design plant. In the flex simulation model, the rate gyro measurements are body rates and they include structural flexibility. A transformation is therefore used, shown in Fig.4, to convert the rate signals to LVLH rates for feedback. The transformation requires the LVLH attitude and the orbital rate $\omega_0=0.0011$ (rad/sec). The model also includes the CMG dynamics implemented as second order transfer functions and the aerodynamic disturbance torques in roll, pitch, and yaw. There is also a bias torque as shown in Equation 1.

```

% LQR Design for Gravity Gradient Momentum Management -----
start;
[A1, B1, C1, D1]= lvlh_plant;           % Load the LVLH Design Plant Model
[Ag, Bg, Cg, Dg]= augm_plant;         % Load the Augmented Design Plant Model
[A0,B0,C0,D0]=linmod('Augm_Vehicle'); % Create State-Space system for ...
sys=ss(A0,B0,C0,D0);                 % Analyze the Augmented System
sys=ss(Ag,Bg,Cg,Dg);                 % Analyze the Augmented System
Q0=diag([100, 1.e+6, 100, ...        % lvlh attitude weights
         0.1, 0.1, 0.1, ...         % lvlh rate weights
         1.e-10, 1.e-11, 1.e-10, ... % CMG Momentum weights
         1.e-12, 3.e-14, 1.e-12]);  % CMG Moment-Integral weights
R0=diag([0.3, 1, 0.1]*1.e-3);       % CMG Control torque Weights
[Kpqr,S,E] = lqr(sys,Q0,R0)
save Kpqr.mat Kpqr -ascii           % Save the LQR gains in Kpqr.mat
[A1,B1,C1,D1]=linmod('Simple_Sim'); % Create State-Space system for ...

```

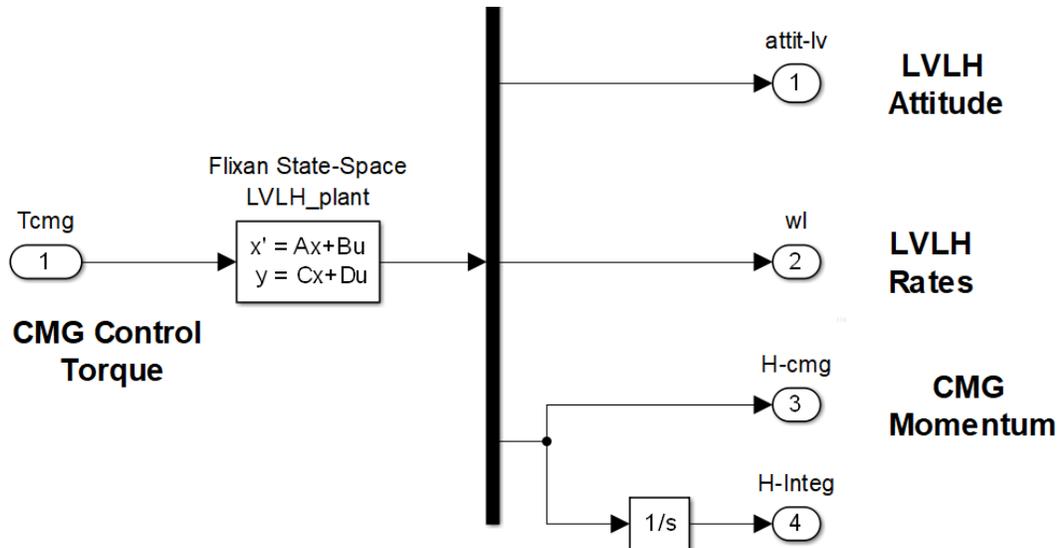


Figure 2 Augmented 12 State Plant Model used for LQR Design including the Momentum Integral

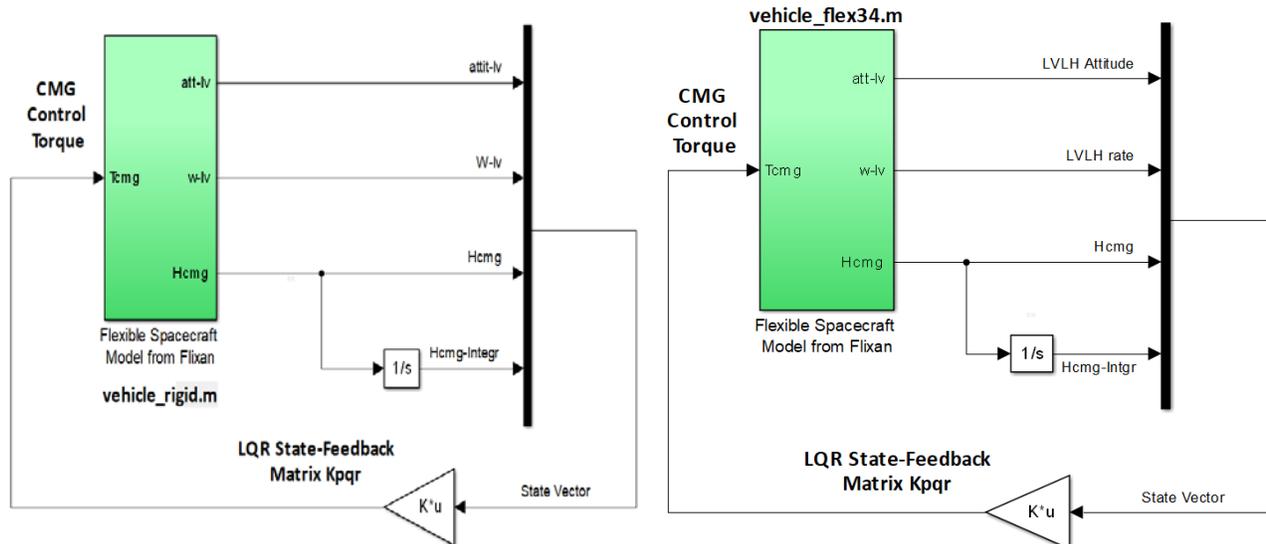
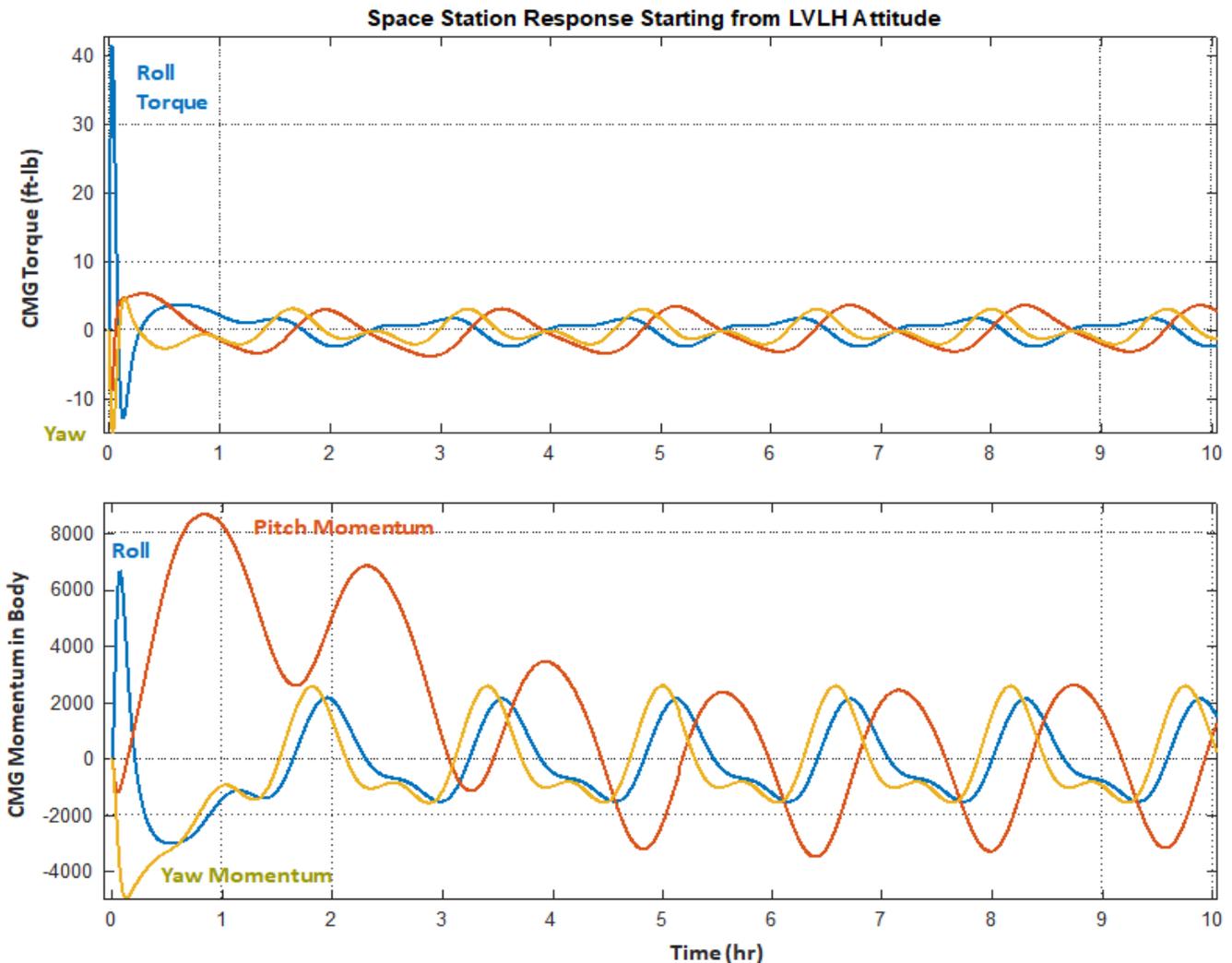


Figure 3 Rigid and Flex Closed Loop Simulation Models with State-Feedback K_{pqr}

3.3 Simulation Results

We will now use the simulation model “*Sim_Lin_TEA_Flex.mdl*” to calculate the spacecraft response to aerodynamic disturbances. The disturbance torques are shown in Figure 5 and they consists of cyclic and steady components. They are included in the Space Station dynamics block, shown in Figure 4. There is no attitude command in the system when the Station is operating in this mode. The Space Station attitude is initialized at zero, that is, equal to the LVLH attitude, and it simply drifts under the influence of the external aerodynamic torques and also the gravity gradient torques which are included in the equations of motion.

As the momentum begins to grow the spacecraft changes its attitude, drifting towards torque equilibrium (TEA) and it uses gravity-gradient to balance the steady aero moments. The cyclic components of the disturbances, however, generate attitude oscillations, mostly in-plane (pitch). The state-feedback control stabilizes not only the attitude but also manages the CMG momentum and prevents it from diverging. The momentum integral feedback prevents it from being biased, and it is eventually cycling around zero because of the disturbances at ω_0 and at $2\omega_0$. Notice the control system bandwidth cannot be sufficiently opened in order to provide more torque and to reduce the attitude oscillations caused by the disturbances. The CMGs have torque and momentum limitations and the flex modes can be amplified to instability. In the next section we shall include disturbance accommodation filters to further attenuate the attitude oscillations at orbital rate.



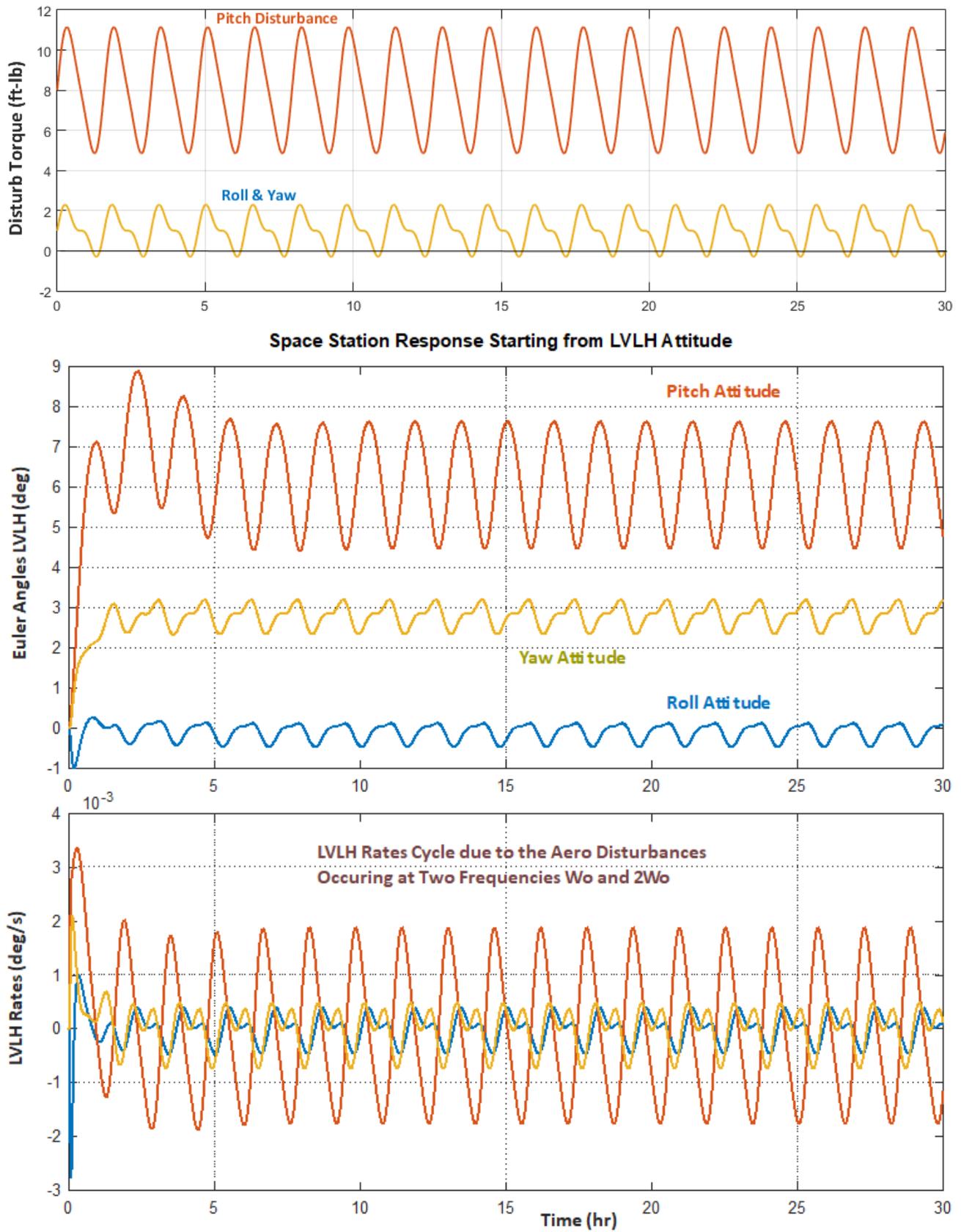


Figure 5 Spacecraft Response to Disturbances Calculated from Simulink Model “Sim_Lin_TEA_Flex.mdl”

3.4 Stability Analysis

The Simulink model “Open_Lin-TEA_Flex.mdl” is used to calculate the frequency responses of the 3 loops system by opening one loop while keeping the other two closed. The loop is opened at the CMG control torque. It is shown in Figure 6 configured for pitch analysis. The Matlab script m-file “freq.m” is using this model to calculate the Bode and Nichols plots.

Figures 7, 8 and 9 show the stability analysis results obtained from this model, one loop at a time. Structural flexibility becomes an issue if we try to open up the bandwidth too much and we can, therefore, offer only limited amount of disturbance attenuation.

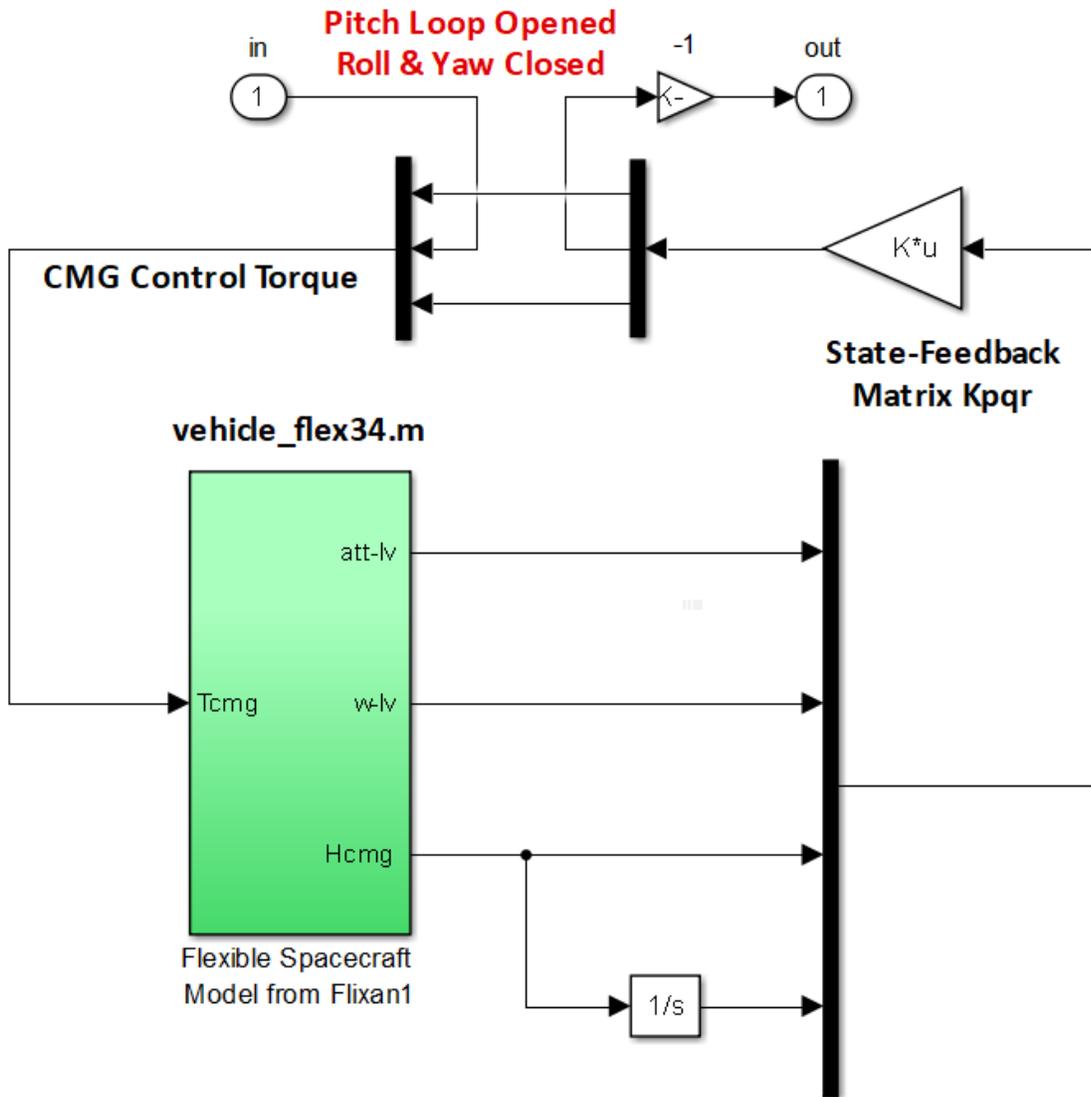


Figure 6 Simulink Model “Open_Lin-TEA_Flex.mdl” Used to Calculate the Frequency Responses and Analyze Stability

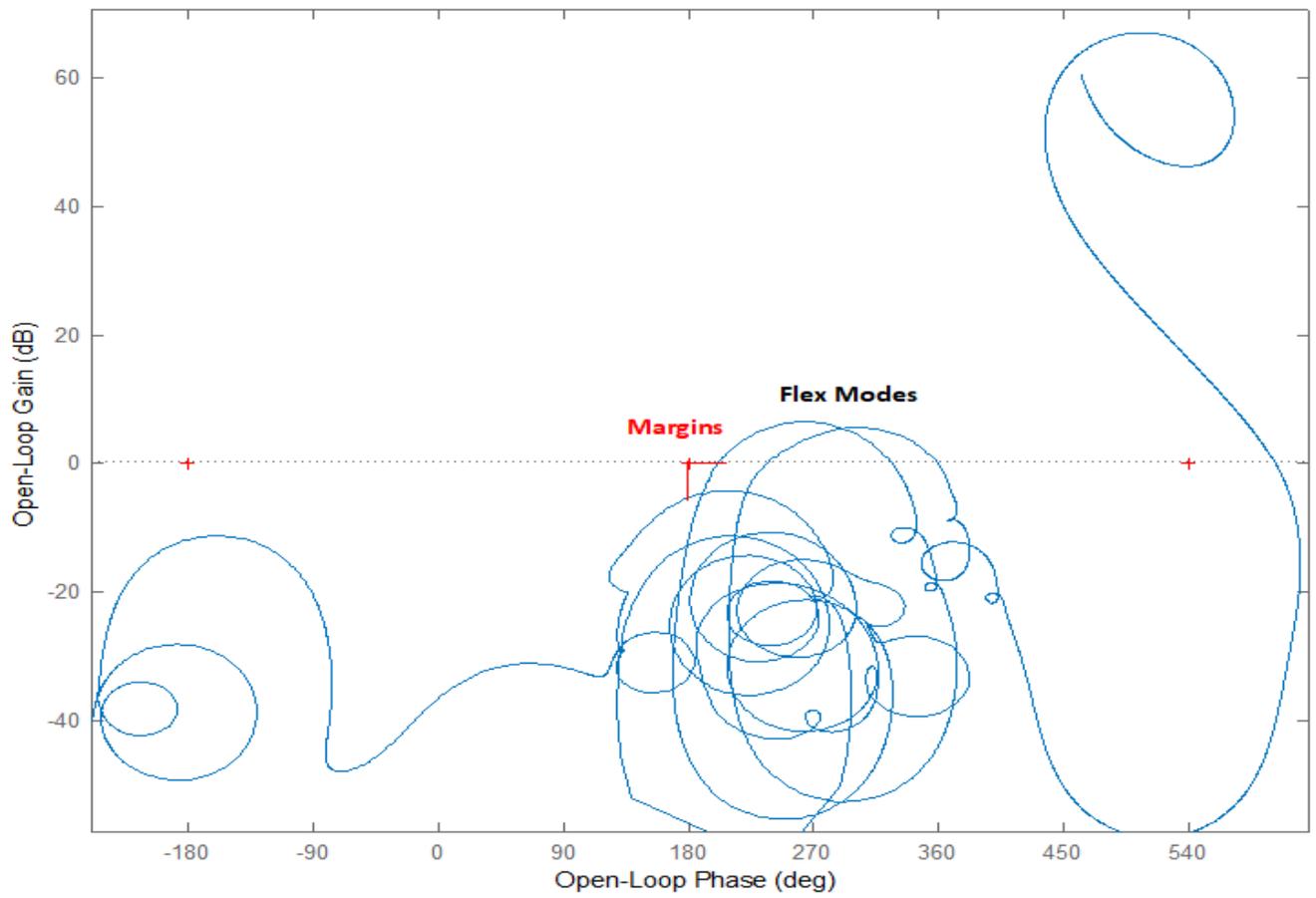
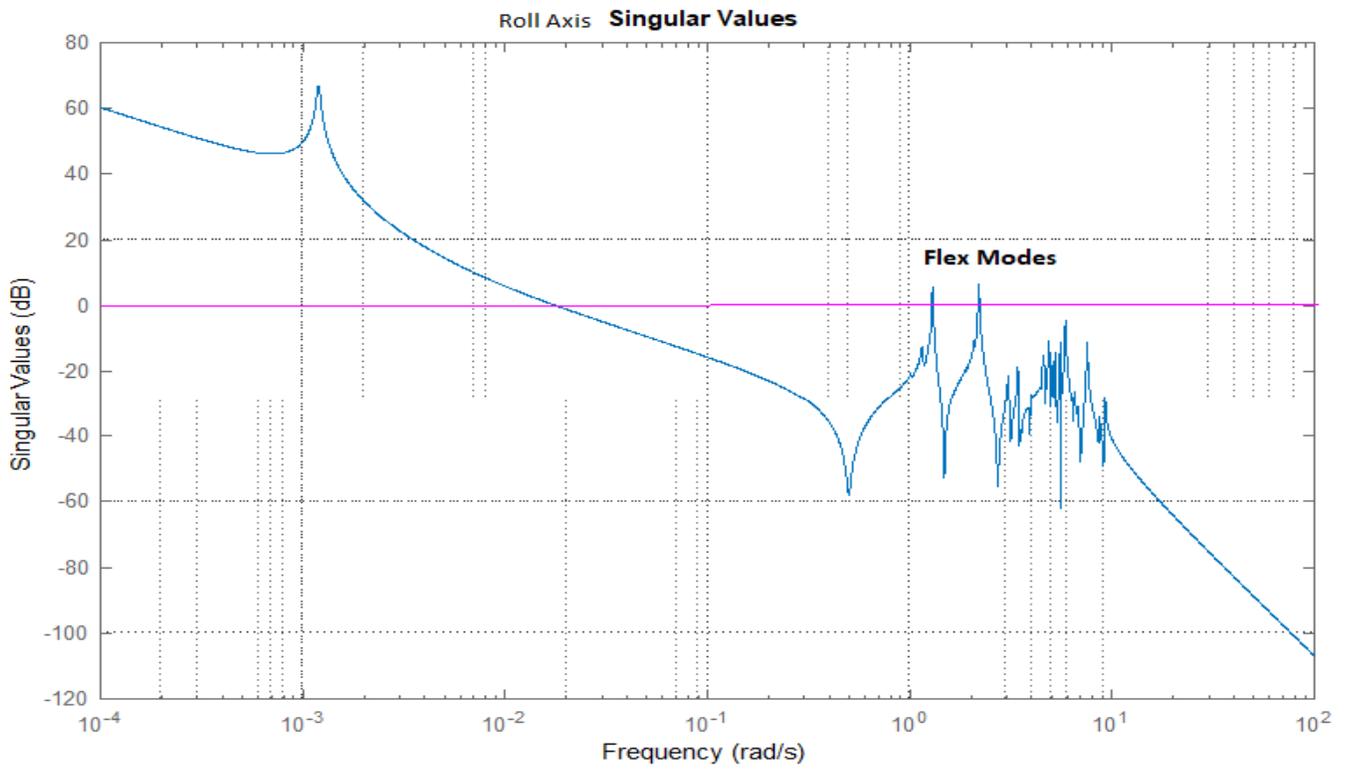


Figure 7 Roll Axis Bode and Nichols Plots showing Flexibility and Stability Margins

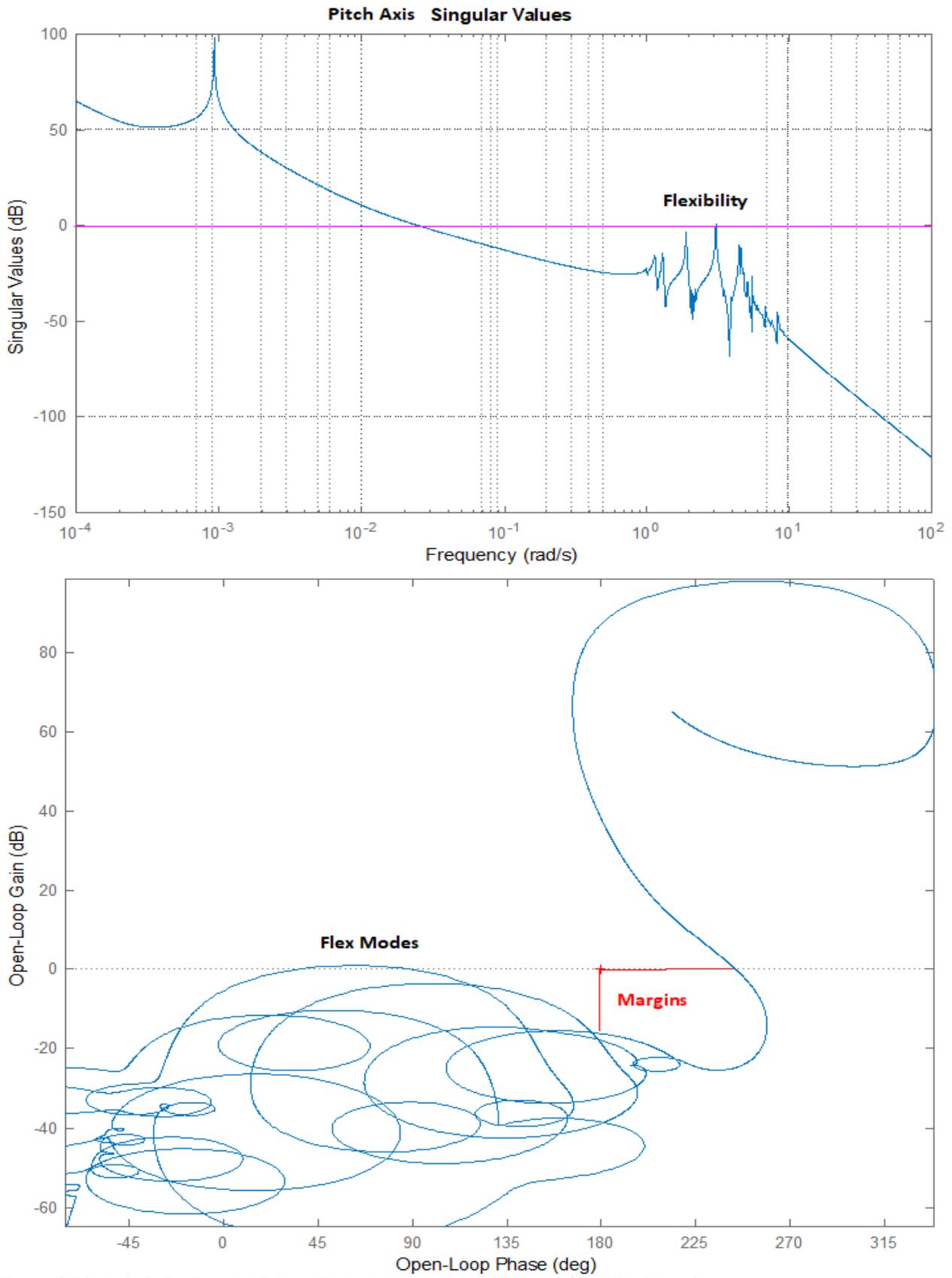


Figure 8 Pitch Axis Bode and Nichols Plots showing Flexibility and Stability Margins

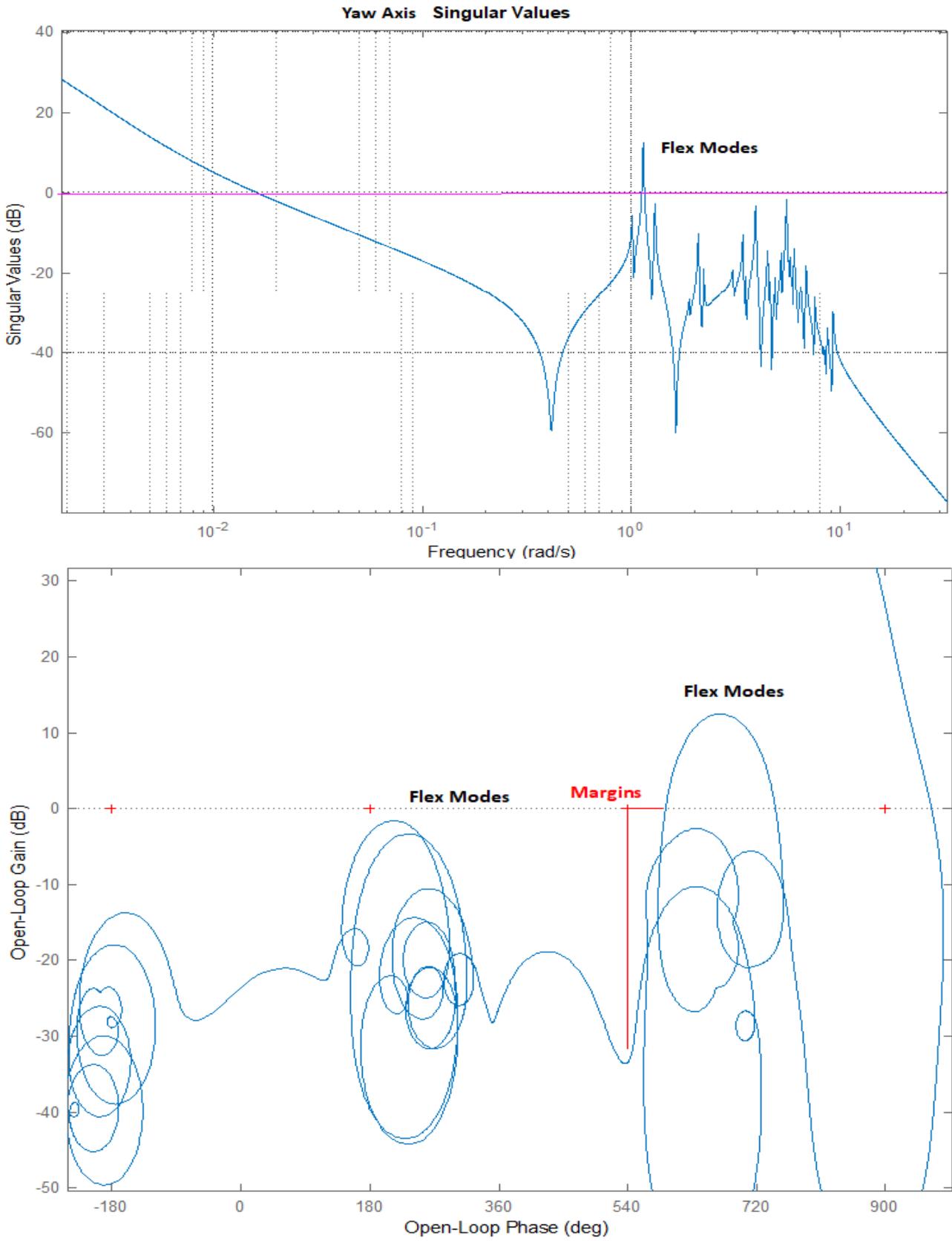


Figure 9 Yaw Axis Bode and Nichols Plots showing Flexibility and Stability Margins

4. Design with Disturbance Attenuation Filters

The cyclic disturbances occurring at orbital rate are big enough to produce sizeable attitude oscillations at orbital frequency ω_0 , especially in pitch. Since we know the exact frequency of the disturbance, a more efficient design approach is to attenuate its effect on the attitude by introducing a resonance in the design system at the same frequency. The resonance is implemented by including two additional states (α_1, α_2) which are excited by the pitch attitude oscillations θ . A similar resonance with states (β_1, β_2) is introduced in yaw and it is excited by the yaw attitude oscillations ψ . The roll axis doesn't need it as much. When these states are properly penalized in the LQR optimization they will provide further attenuation in the pitch and yaw oscillations. We are essentially increasing the effectiveness of the control system at a certain frequency in order to attenuate known disturbances.

4.1 Flixan Models

The files for this analysis are in directory: "*Flixan\Control Analysis\LQG\Examples\Space-Station w CMG2\Design-2*". The Space Station parameters are defined in the input file "*Space_Station.Inp*" which includes two flight vehicle datasets: a rigid-body "*Space Station with Double-Gimbal CMG Array (Rigid)*", and a flexible model with 34 structural modes "*Space Station with Double-Gimbal CMG Array (Flex)*". The already selected and scaled modes are in dataset "*Space Station with Double-Gimbal CMG Array, 34 Flex Modes*". The "**LVLH Attitude & Rate**" flag is included in the flags line that defines the output attitude and rate relative to the LVLH frame. The rate states however are still in body. A batch set is included at the top of the input file with title "*Batch for Large Flexible Space Station*" and it is used for processing the entire input file fast. The systems and matrices created by the program are saved in file "*Space-Station.Qdr*".

```
BATCH MODE INSTRUCTIONS .....
Batch for Large Flexible Space Station
! This batch set creates a models for a Space Station that is
! controlled by an array of double-gimbal CMGs. Two models
! are created, a rigid-body model and a flexible model using the
! attached modal data. The design model is extracted from Rigid Vehicle
! and the plant state is transformed so that it is equal to the Output, C=I
!
Retain Matrix      : State Weight Matrix Qc (16x16)
Retain Matrix      : Control Weight Matrix Rc (3x3)
!
Flight Vehicle     : Space Station with Double-Gimbal CMG Array (Rigid)
Flight Vehicle     : Space Station with Double-Gimbal CMG Array (Flex)
System Modificat   : Space Station with Double-Gimbal CMG Array (Design Plant)
System Modificat   : Space Station with Double-Gimbal CMG Array (LVLH Plant)
Transf-Function    : Oscillation Filter
Transf-Function    : 3 Integrators
System Connection  : Augmented Design Plant
LQR Control Des    : LQR Control Design for Augmented Space Station Model
!
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Rigid)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Flex)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (Design Plant)
To Matlab Format    : Space Station with Double-Gimbal CMG Array (LVLH Plant)
To Matlab Format    : Augmented Design Plant
To Matlab Format    : LQR State-Feedback Gain for Augmented Design Plant
-----
```

```

CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)
Space Station with Double-Gimbal CMG Array (Design Plant)
Space Station with Double-Gimbal CMG Array (Rigid)
! Create the Station Design Model by Reducing the Rigid-Body Model
! Inputs are the CMG Control Torques
! Outputs are: LVLH attitude and rates, and CMG Momentum
!
TRUNCATE OR REORDER THE SYSTEM INPUTS, STATES, AND OUTPUTS
Extract Inputs :   1   2   3
Extract States :   1   3   5   2   4   6  11  12  13
Extract Outputs:   1   3   5   2   4   6  16  17  18
-----
CREATE A NEW SYSTEM FROM AN OLD SYSTEM... (Titles of the New and Old Systems)
Space Station with Double-Gimbal CMG Array (LVLH Plant)
Space Station with Double-Gimbal CMG Array (Design Plant)
! Transform the design plant and make the States equal to the Outputs
SYSTEM TRANSFORMATION, STATES EQUAL TO OUTPUTS
-----

```

The input file includes a system modification dataset “*Space Station with Double-Gimbal CMG Array (Design Plant)*” that creates the control design model by extracting a reduced number of inputs, states, and outputs from the rigid model. An additional transformation is needed because although the design plant attitude and rates are in the LVLH frame, some of the states, however, are still in the body frame. This transformation modifies the design plant and makes the states to be equal to the outputs which are LVLH attitudes and rates and the output matrix $C = \text{Identity}$. The dataset that performs this transformation is “*Space Station with Double-Gimbal CMG Array (LVLH Plant)*”. Notice that the transformation requires the design plant matrix C to be square and invertible.

SYSTEM OF TRANSFER FUNCTIONS ...

Oscillation Filter

! This Second Order Filter Amplifies the Oscillation Disturbance
! for the LQR Optimization

Continuous

```

TF. Block # 1 Integrator a1 Order of Numer, Denom= 0 1
Numer 0.0 1.0
Denom 1.0 0.0
TF. Block # 2 Integrator a2 Order of Numer, Denom= 0 1
Numer 0.0 1.0
Denom 1.0 0.0

```

```

.....
Block #, from Input #, Gain
1 1 1.0
.....
Block #, from Block #, Gain
2 1 1.0
1 2 -1.21e-6
.....

```

```

.....
Outpt #, from Block #, Gain
1 1 1.0
2 2 1.0
.....

```

```

.....
Definitions of Inputs = 1
Pitch Attitude (rad)

```

```

.....
Definitions of Outputs = 2
Filter Output a1
Filter Output a2
-----

```

$$\begin{cases} \dot{\alpha}_1 = \theta - \omega_o^2 \alpha_2 \\ \dot{\alpha}_2 = \alpha_1 \end{cases}$$

The oscillation filter is a second order resonance that is tuned at the disturbance frequency ω_0 . It creates two additional states (α_1, α_2) which are used to penalize attitude oscillations at that frequency, in the LQR optimization. It is implemented as shown above by two integrators combined using the Flixdan transfer functions combination utility.

SYSTEM OF TRANSFER FUNCTIONS ...

3 Integrators

! Used to calculate the CMG Momentum Integral for the LQR Optimization

Continuous

TF. Block #	1	Integrator x	Order of Numer, Denom=	0	1
Numer	0.0	1.0			
Denom	1.0	0.0			
TF. Block #	2	Integrator y	Order of Numer, Denom=	0	1
Numer	0.0	1.0			
Denom	1.0	0.0			
TF. Block #	3	Integrator z	Order of Numer, Denom=	0	1
Numer	0.0	1.0			
Denom	1.0	0.0			

.....
Block #, from Input #, Gain
1 1 1.0
2 2 1.0
3 3 1.0
.....

.....
Outpt #, from Block #, Gain
1 1 1.0
2 2 1.0
3 3 1.0
.....

.....
Definitions of Inputs = 3
CMG Momentum-X
CMG Momentum-Y
CMG Momentum-Z

.....
Definitions of Outputs = 3
CMG Momentum-X Integral
CMG Momentum-Y Integral
CMG Momentum-Z Integral

The design model is now augmented by including 3 additional states, the (x, y, z) momentum integral in order to bound the CMG momentum from diverging and to keep it cycling near zero. Two attitude oscillation filters are also included in the augmented system to further attenuate the pitch and yaw attitude oscillations at orbital frequency. The momentum integrators are implemented as 3 transfer functions system by the transfer functions combination utility, shown above. The “*Augmented Design Plant*” is obtained by combining four systems as shown in Figure 10 using the Flixan systems combination utility. It is also implemented in the Simulink file “*Augm_Vehicle.mdl*”. In addition to the 9 states of the original LVLH design plant, the augmented plant now includes the 3 momentum integral states and the four attitude filter states (2 pitch and 2 yaw).

INTERCONNECTION OF SYSTEMS

Augmented Design Plant

! Augment the Design Plant by including the Momentum Integral and the
! Oscillation Filters

Titles of Systems to be Combined

Title 1 Space Station with Double-Gimbal CMG Array (LVLH Plant)

Title 2 3 Integrators

Title 3 Oscillation Filter

Title 4 Oscillation Filter

SYSTEM INPUTS TO SUBSYSTEM 1

Via Matrix +I3

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 1

System Output 1 from Subsystem 1, Output 1, Gain= 1.0

System Output 2 from Subsystem 1, Output 2, Gain= 1.0

System Output 3 from Subsystem 1, Output 3, Gain= 1.0

System Output 4 from Subsystem 1, Output 4, Gain= 1.0

System Output 5 from Subsystem 1, Output 5, Gain= 1.0

System Output 6 from Subsystem 1, Output 6, Gain= 1.0

System Output 7 from Subsystem 1, Output 7, Gain= 1.0

System Output 8 from Subsystem 1, Output 8, Gain= 1.0

System Output 9 from Subsystem 1, Output 9, Gain= 1.0

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 2

System Output 10 from Subsystem 2, Output 1, Gain= 1.0

System Output 11 from Subsystem 2, Output 2, Gain= 1.0

System Output 12 from Subsystem 2, Output 3, Gain= 1.0

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 3

System Output 13 from Subsystem 3, Output 1, Gain= 1.0

System Output 14 from Subsystem 3, Output 2, Gain= 1.0

.....

SYSTEM OUTPUTS FROM SUBSYSTEM 4

System Output 15 from Subsystem 4, Output 1, Gain= 1.0

System Output 16 from Subsystem 4, Output 2, Gain= 1.0

.....

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 2

Subsystem 1, Output 7 to Subsystem 2, Input 1, Gain= 1.0

Subsystem 1, Output 8 to Subsystem 2, Input 2, Gain= 1.0

Subsystem 1, Output 9 to Subsystem 2, Input 3, Gain= 1.0

.....

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 3

Subsystem 1, Output 2 to Subsystem 3, Input 1, Gain= 1.0

.....

SUBSYSTEM NO 1 GOES TO SUBSYSTEM NO 4

Subsystem 1, Output 3 to Subsystem 4, Input 1, Gain= 1.0

.....

Definitions of Inputs = 3

Roll CMG Momentum (ft-lb)

Ptch CMG Momentum (ft-lb)

Yaw CMG Momentum (ft-lb)

Definitions of Outputs = 16

Roll Attitude (phi-LVLH) (radians)

Pitch Attitude (thet-LVLH) (radians)

Yaw Attitude (psi-LVLH) (radians)

Roll Rate (p-lvlh) (rad/sec)

Pitch Rate (q-lvlh) (rad/sec)

Yaw Rate (r-lvlh) (rad/sec)

CMG Momentum in X-axis (ft-lb-sec)

CMG Momentum in Y-axis (ft-lb-sec)

CMG Momentum in Z-axis (ft-lb-sec)

CMG Momentum Integral X (ft-lb-sec^2)

CMG Momentum Integral Y (ft-lb-sec^2)

CMG Momentum Integral Z (ft-lb-sec^2)

Pitch Oscillation Filter Output (a1)

Pitch Oscillation Filter Output (a2)

Yaw Oscillation Filter Output (b1)

Yaw Oscillation Filter Output (b2)

Pitch Filter

Yaw Filter

3 CMG Torques

9 Original States

3 LVLH Attitudes

3 LVLH Rates

3 CMG Momentum

from Moment Integr

H-integr-X

H-integr-Y

H-integr-Z

from Pitch Filter

a1

a2

from Yaw Filter

b1

b2

to Momen Integrat

CMG Momentm-X

CMG Momentm-Y

CMG Momentm-Z

to Pitch Filter

Pitch Attitude

to Yaw Filter

Yaw Attitude

The above dataset contains system interconnection instructions for implementing the augmented plant model shown in Figure 10.

The next dataset included in the input file is the “*LQR Control Design for Augmented Space Station Model*” which calculates the (3x16) state-feedback matrix K_{pqr} . The state and control weight matrices Q_c and R_c are already included in file “*Space_Station.Qdr*”. The state-feedback matrix K_{pqr} is also saved in file “*Space_Station.Qdr*” and its title is “*LQR State-Feedback Gain for Augmented Design Plant*”.

```

LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design for Augmented Space Station Model
Plant Model Used to Design the Control System from:      Augmented Design Plant
Criteria Optimization Output is Matrix C
State Penalty Weight (Qc) is Matrix: Qc16              State Weight Matrix Qc (16x16)
Control Penalty Weight (Rc) is Matrix: Rc3             Control Weight Matrix Rc (3x3)
Continuous LQR Solution Using Asymptotic Method
LQR State-Feedback Control Gain Matrix Kpqr            LQR State-Feedback Gain for Augmented Design Plant
-----

```

Six Matlab conversion datasets are also included that convert the systems and matrices in an m-file format that can be loaded into Matlab by running the script file start.m. The files “vehicle_rigid.m” and “vehicle_flex34.m” are used in simulations. The augmented plant “augm_plant.m” can also be used in Matlab to calculate the state-feedback matrix K_{pqr} .

```

CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Rigid)
System
vehicle_rigid
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Flex)
System
vehicle_flex34
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (Design Plant)
System
design_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Space Station with Double-Gimbal CMG Array (LVLH Plant)
System
lvlh_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
Augmented Design Plant
System
augm_plant
-----
CONVERT TO MATLAB FORMAT ..... (Title, System/Matrix, m-filename)
LQR State-Feedback Gain for Augmented Design Plant
Matrix Kpqr
-----

```

4.2 LQR Control Design with Filters

The initialization file “start.m” loads the dynamic systems and matrices for the simulation and analysis models. In addition to Flixan, the Matlab script file “des.m” can also calculate the LQR control gains. It is using the augmented design plant or the Simulink model “Augm_Vehicle.mdl” shown in Figure 10, which includes the spacecraft LVLH dynamics of Equation 1, the momentum integral states, and the two attitude filters. The state vector of the augmented design plant includes the 4 filter states (α_1 , α_2 , β_1 , β_2) and the augmented 16-state system is now applied in the LQR optimization algorithm. The disturbance accommodation filters will penalize the attitude oscillations at orbital rate and ultimately provide further attenuation at that frequency.

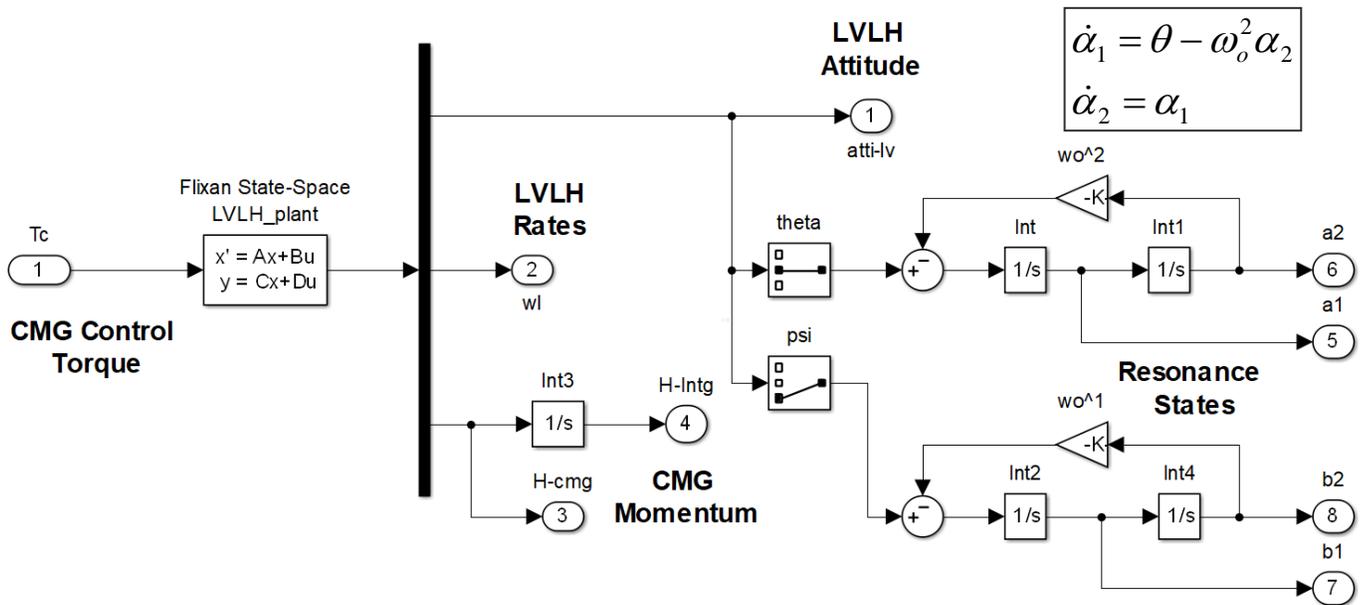


Figure 10 Augmented Design Plant “Augm_Vehicle” Consists of Equations 1, Momentum Integrals and the Pitch and Yaw Attitude Augmentation States (α_1, α_2) and (β_1, β_2)

Figure 11 shows the simulation model “*Sim_Lin_TEA-Flex.mdl*” that includes the Flixan generated system “*vehicle_flex34.m*” with 34 flex modes. It includes also the two second order attitude augmentation filters which are now part of the control system and produce the 4 additional states: ($\alpha_1, \alpha_2, \beta_1, \beta_2$) required for state-feedback. The rate gyro measurements from the dynamic model are body rates and they measure structural flexibility. A transformation is used to convert the rate signals to LVLH rates as expected for feedback. The spacecraft model also includes the CMG dynamics implemented as second order transfer functions and the aerodynamic disturbances, as in Fig.4.

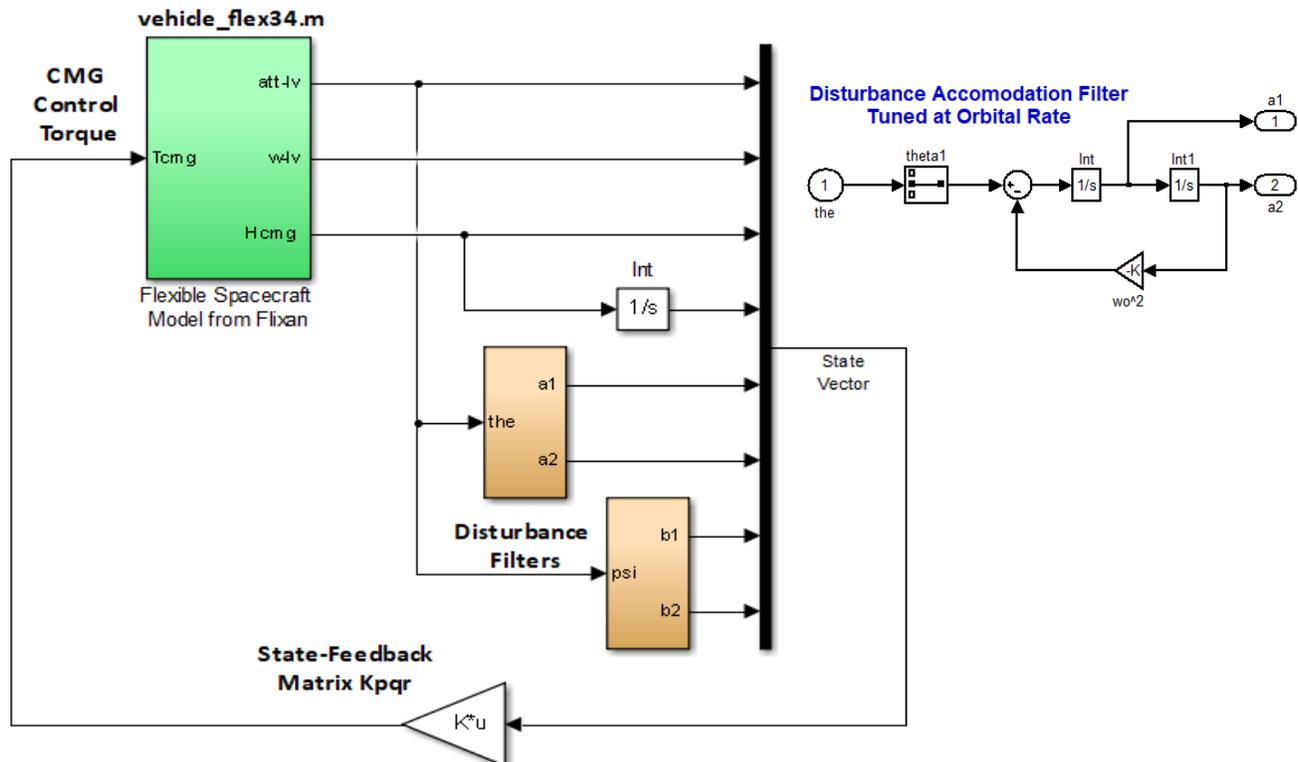
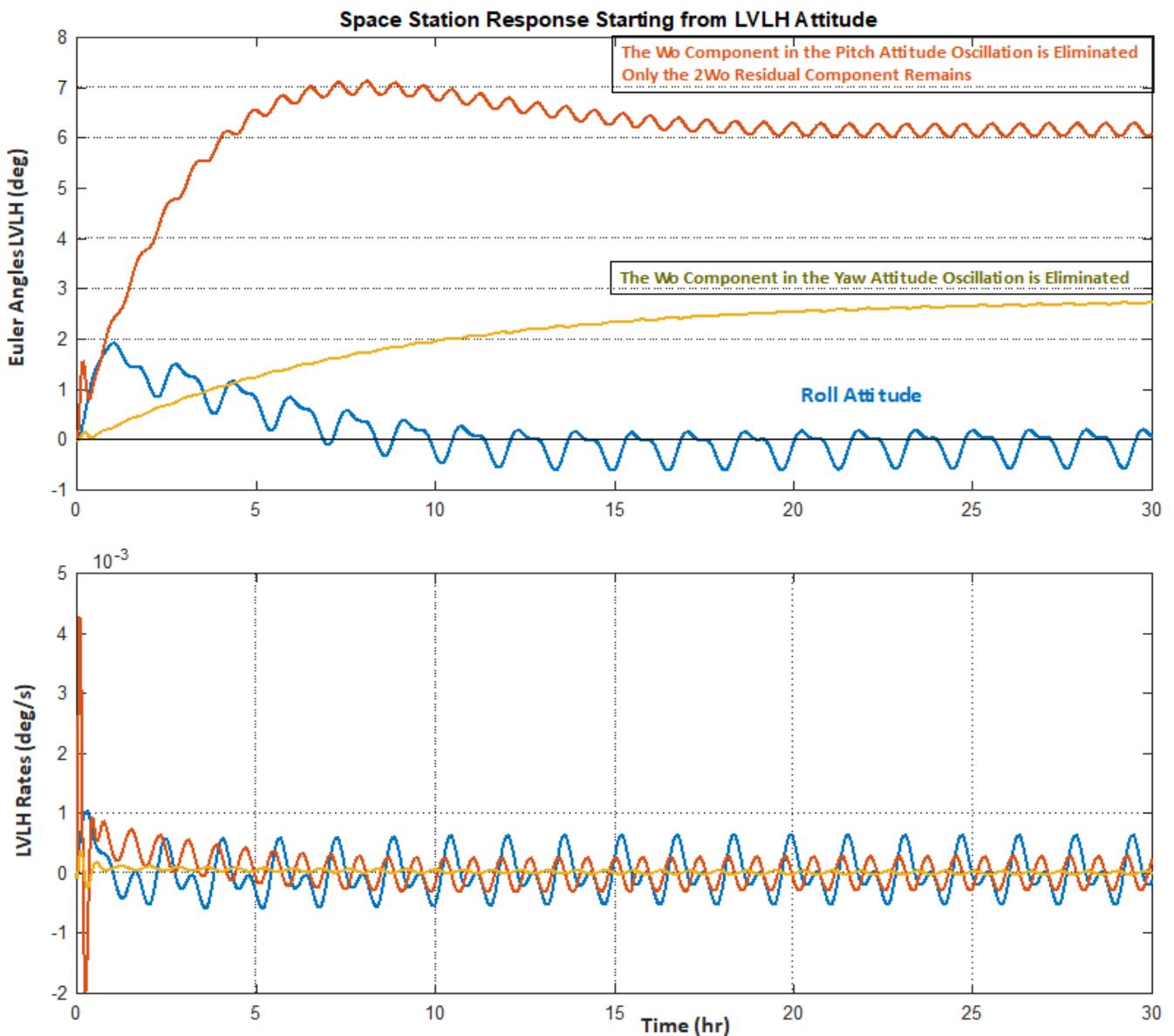


Figure 11 Flex Simulation Model “*Sim_Lin_TEA-Flex.mdl*”

4.3 Simulation Results

The simulation results in Figure 12 show the Space Station response to the aerodynamic disturbances as its attitude drifts towards the TEA where the external torques balance in all directions. At steady-state the pitch attitude converges to 6.3° and the yaw attitude to 3° relative to the LVLH frame. The roll attitude is small at -0.5° . The CMG momentum does not diverge but it oscillates about zero without bias as the CMGs are supplying torque to counteract the cyclic disturbances. The CMG torque is also centered at around zero. The augmented state-feedback not only stabilizes the spacecraft but it also attenuates the ω_o attitude oscillations in pitch and yaw and produces a closed loop system that is resisting the disturbances at orbital rate without the need to increase the control system bandwidth. In pitch the only remaining oscillation is due to the aero disturbance at $2\omega_o$. The yaw attitude is almost perfectly clean from oscillations. Additional filters can be included to attenuate the $2\omega_o$ frequency but we leave this as an exercise for the reader.



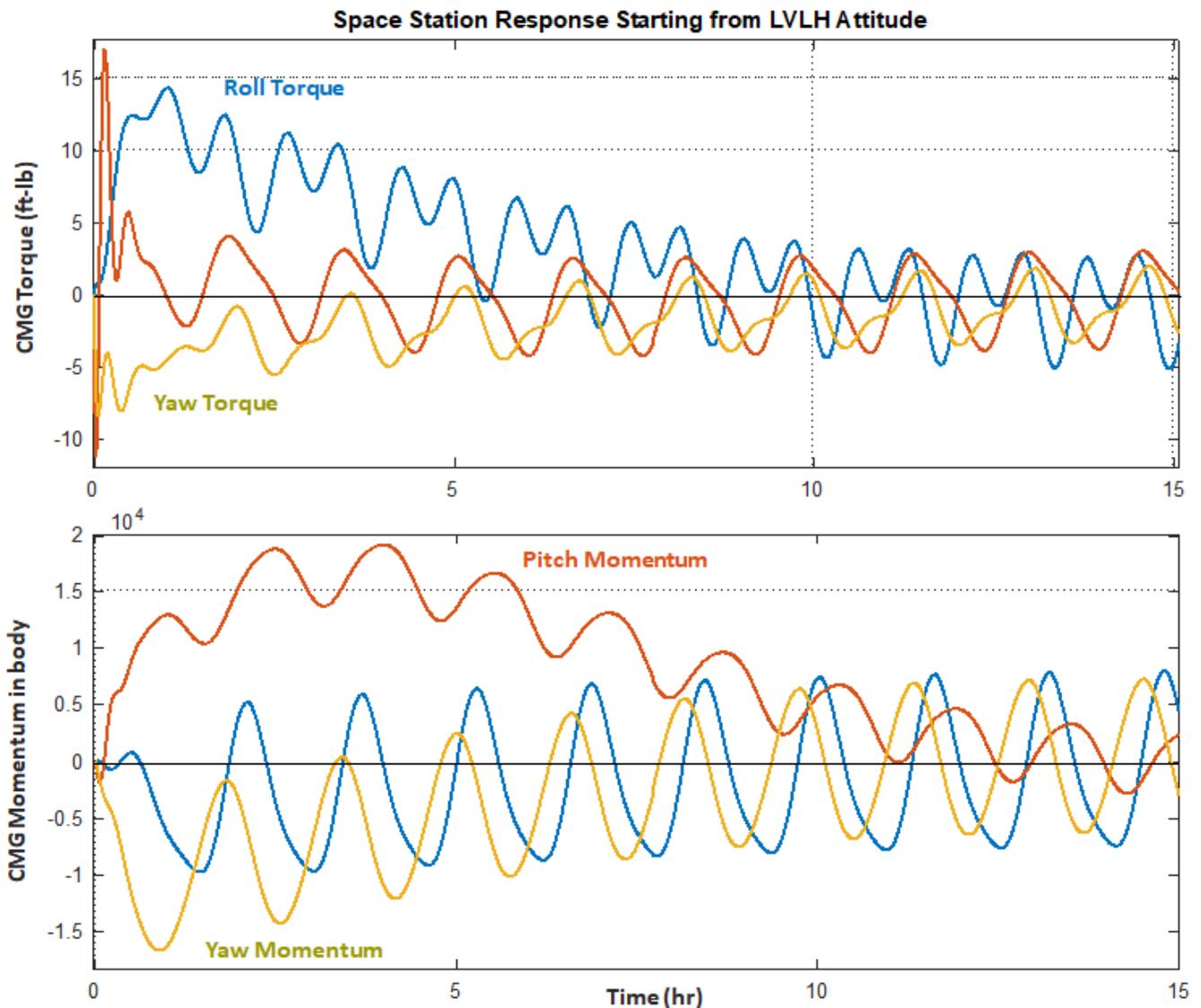


Figure 12 Spacecraft Response to Disturbances from Simulink Model “Sim_Lin_TEA_Flex.mdl”

4.4 Stability Analysis

The Simulink model “Open_Lin_TEA_Flex.mdl” is used to calculate the frequency responses of the 3 loops system by opening one loop while keeping the other two closed. The loop is opened at the CMG control torque. It is shown in Figure 13 configured for roll axis analysis. The Matlab script m-file “freq.m” is using this model to calculate the Bode and Nichols plots.

Figures 14, 15 and 16 show the stability analysis results obtained from the open-loop model, one loop opened at a time. This system has a low bandwidth of $10\omega_0$. Notice that the filters introduced a big resonance at orbital frequency $\omega_0=0.0011$ (rad/sec) which increases the gain of the system only at that frequency. The increased magnitude is needed in order to provide the torque that will counteract the disturbance at that frequency without increasing the overall system gain that would otherwise amplify the flex modes to instability.

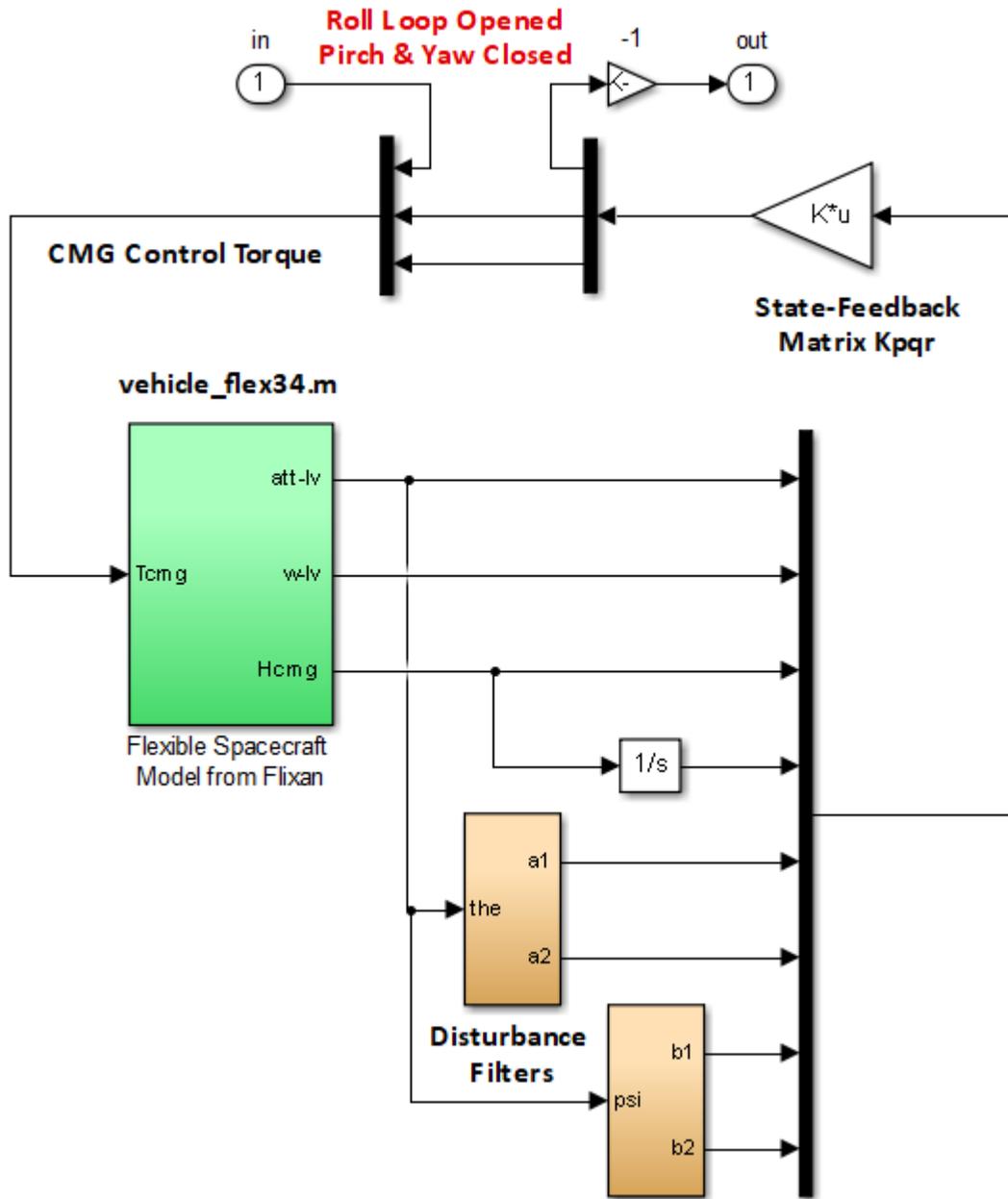


Figure 13 Simulink Model “Open_Lin_TEA_Flex.mdl” Used to Calculate the Frequency Responses and Analyze Stability

5. Conclusion

The LQR method was used to design state-feedback control system for the space station that uses gravity gradient to prevent the CMG momentum from saturating but it is cycling at around zero while it provides the torque necessary to counteract aerodynamic disturbances. The spacecraft attitude is not commanded but it drifts and slightly oscillates about the TEA. Filters are used to amplify the system’s response at the disturbance frequency, counteract the disturbance effect, and to minimize the attitude oscillations.

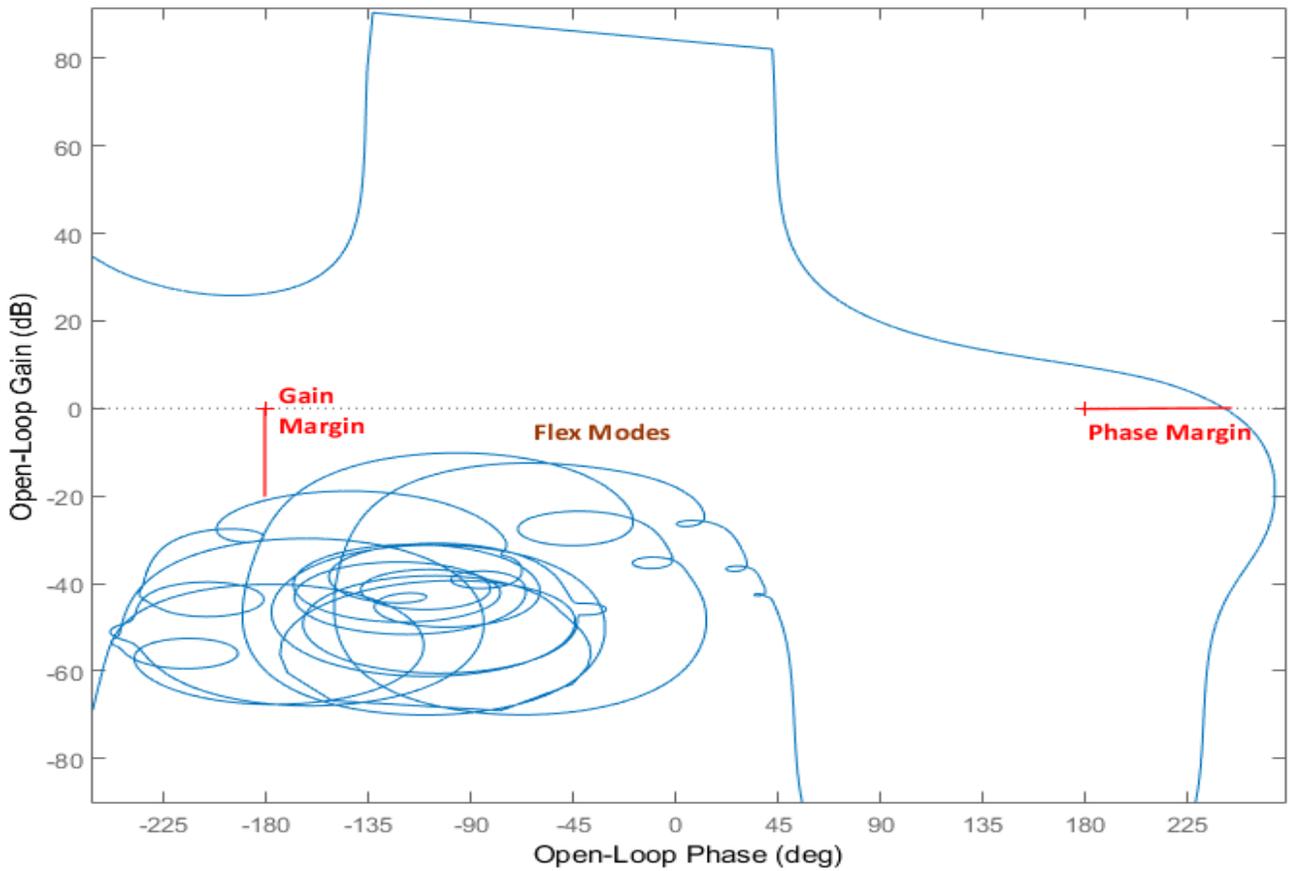
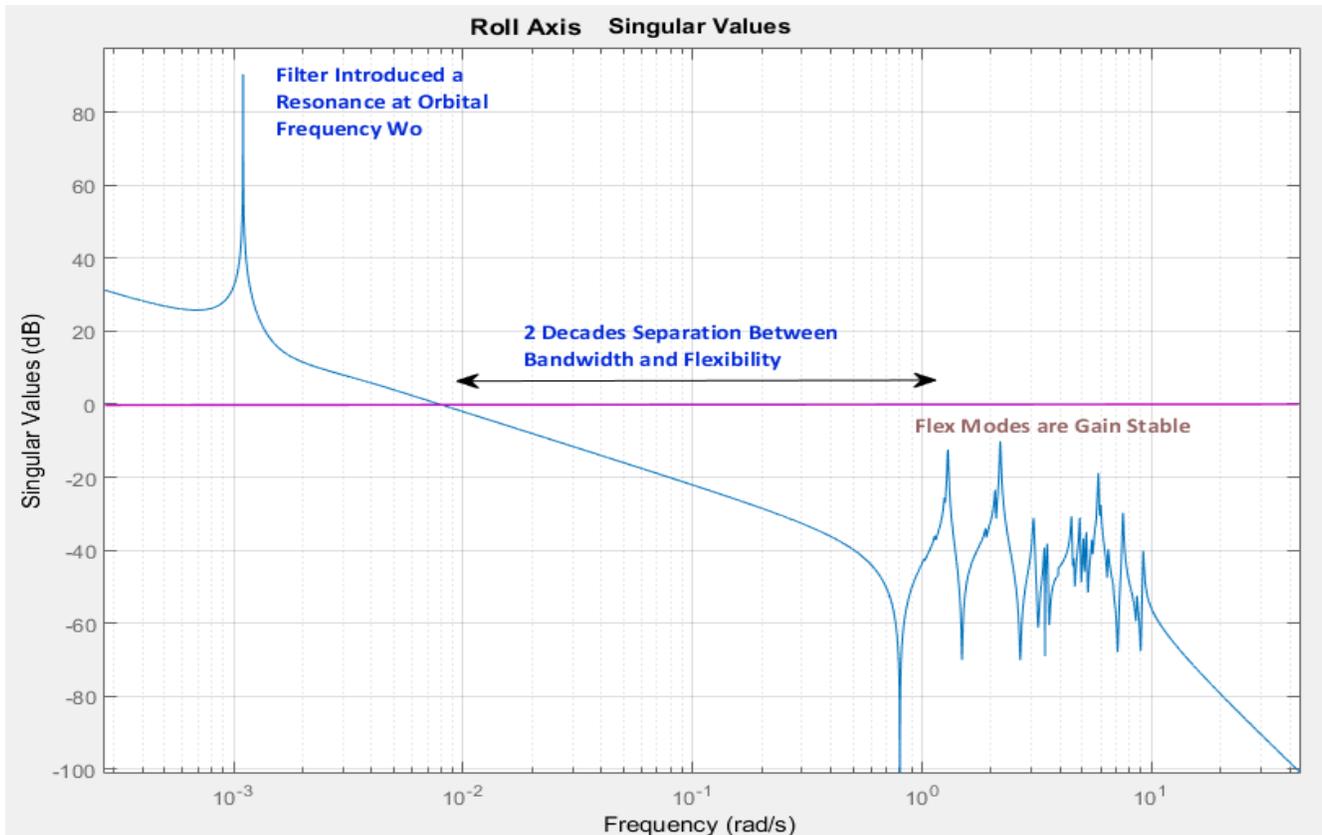


Figure 14 Roll Axis Bode and Nichols Plots showing Flexibility and Stability Margins

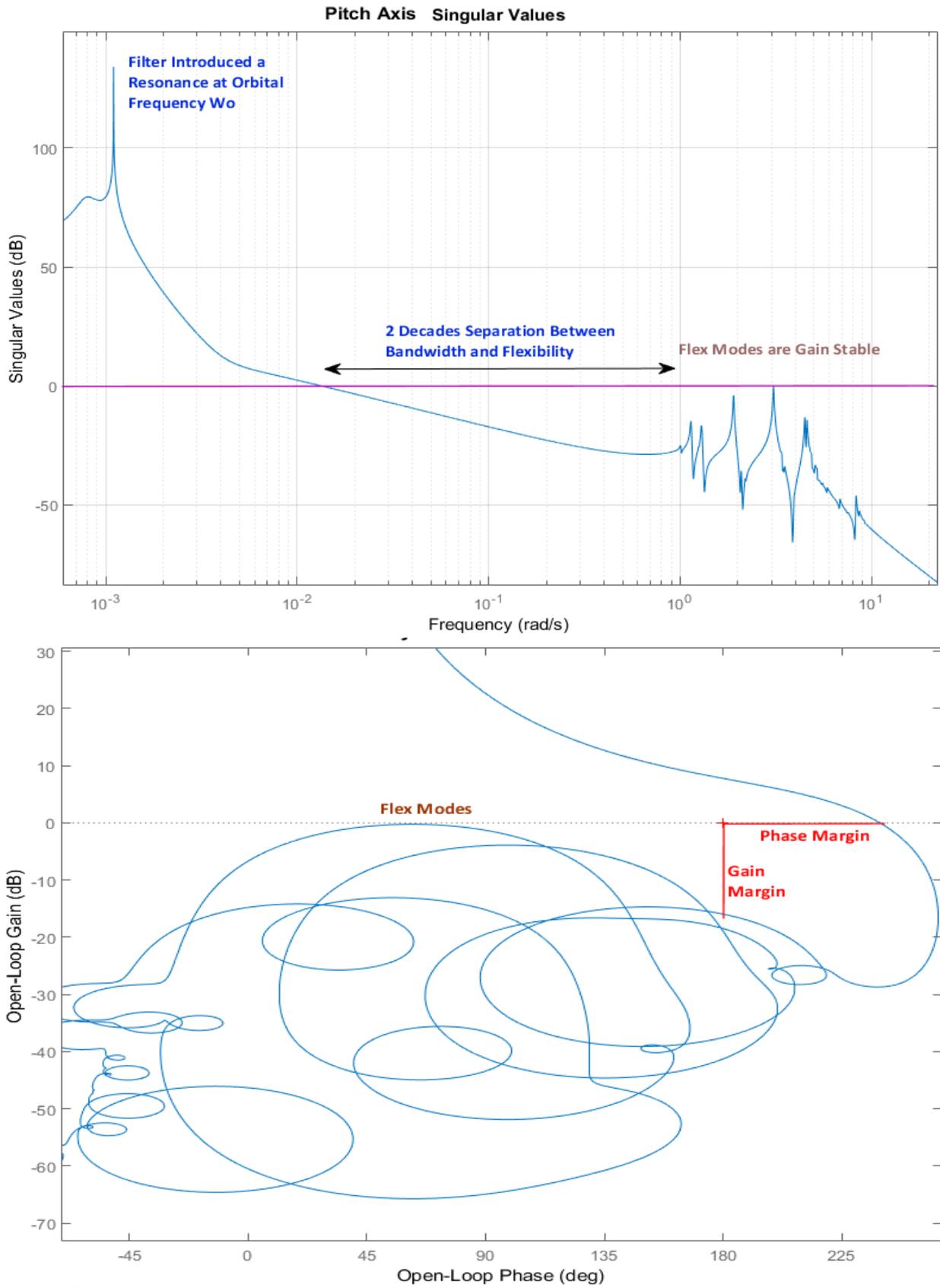


Figure 15 Pitch Axis Bode and Nichols Plots showing Flexibility and Stability Margins

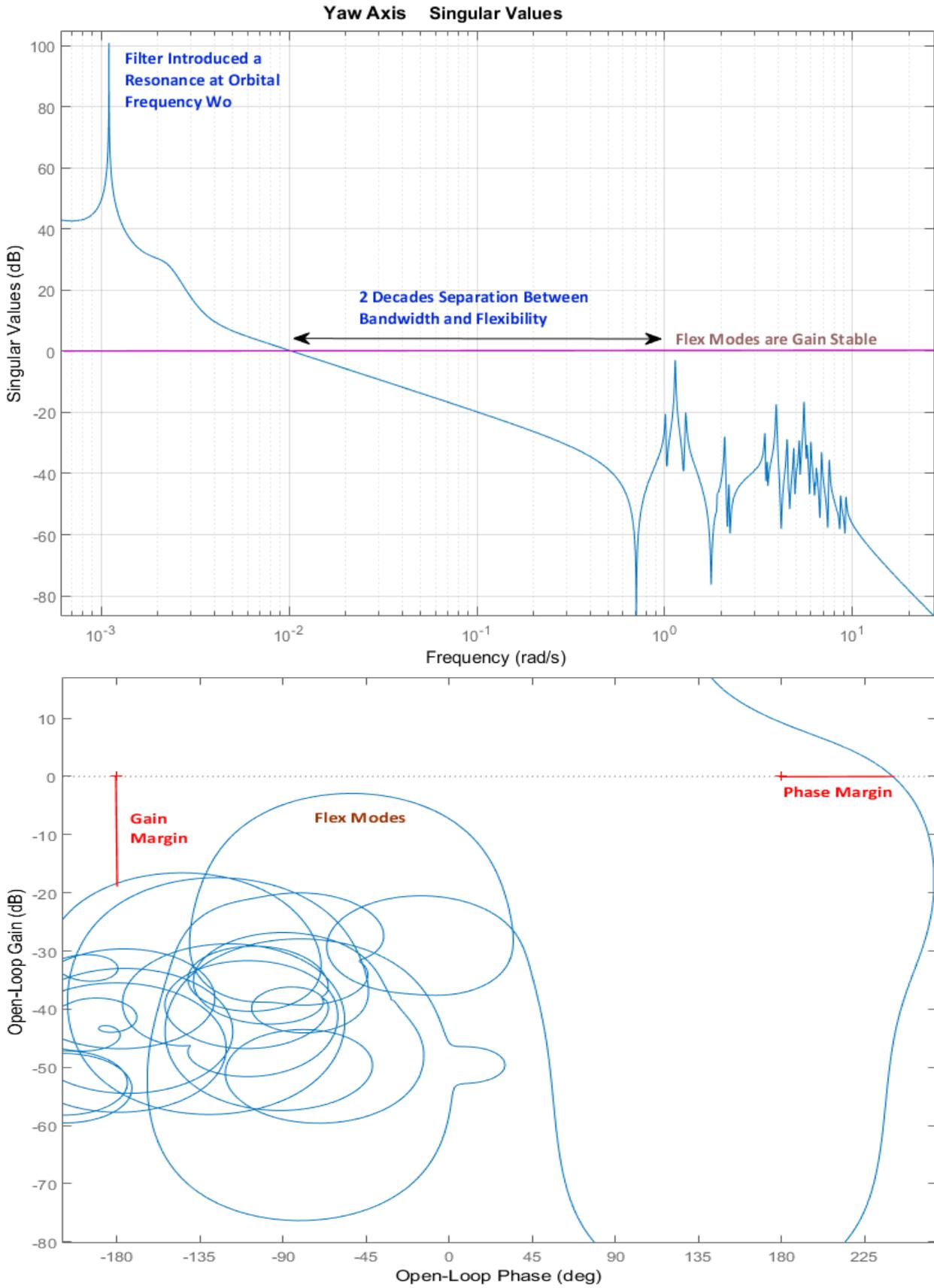


Figure 16 Yaw Axis Bode and Nichols Plots showing Flexibility and Stability Margins

Frequency Response Analysis Program

The frequency response analysis program calculates the frequency response of a system that is described in state-space form by a set of four matrices (A,B,C,D), either continuous or discrete at fixed sampling rate. It can display the frequency responses in Bode, Nichols and Nyquist plots. The systems are read from a typical systems file (*.Qdr) containing multiple systems, and the frequency responses are saved in a file with extension (*.Frq). The program includes many options. One of the options employs a variable frequency step feature for calculating smooth Nichol's and Nyquist plots which is useful in analyzing stability of systems that have very low damped resonances such as structural or slosh modes. It can also overlay multiple curves on the same plot for comparison.

The options are selected graphically using the mouse and menus. Multiple frequency responses can be calculated and saved in the same (.Frq) file for various systems and from different inputs and outputs. The user may point the cursor and read data at specific points on the locus, focus in a smaller area to view details which are otherwise not visible in a larger scale, or expand in a larger area. The program is also used to analyze the existence of limit-cycles in non-linear systems by overlaying the inverse of the describing function of the non-linearity on a Nichols or Nyquist plot.

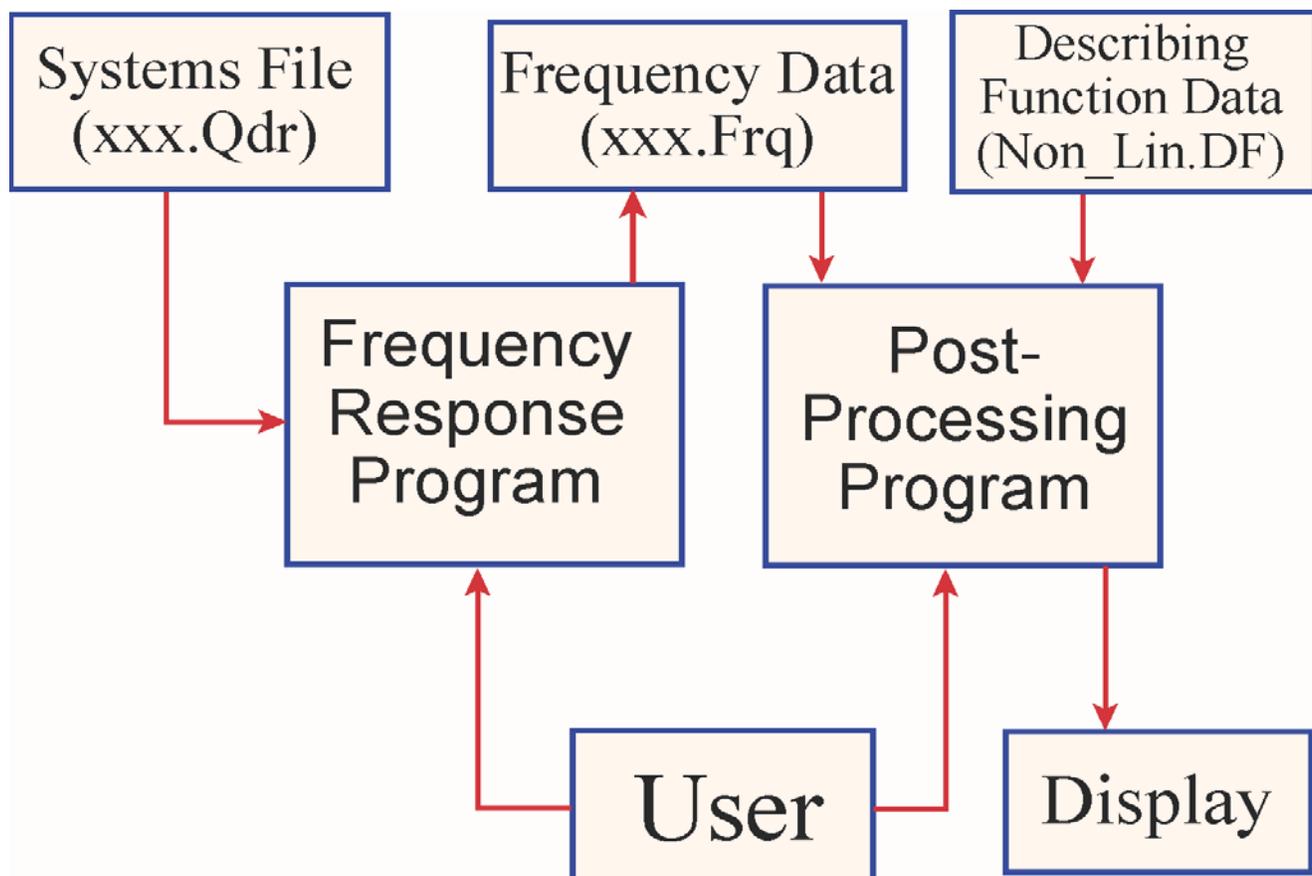


Figure 1 Frequency Response Analysis Program

Program Overview

The frequency response program consists of two parts: the frequency response calculation program, and the graphical analysis program that post-processes the frequency response data and generates different types of plots on the screen.

Frequency Response Calculation: The frequency response calculation program reads the state-space matrices from the systems file (xxx.Qdr), where the filename “xxx” typically describes the flight vehicle. It calculates the frequency responses of more than one system between specific inputs and outputs, and it saves the frequency response data in file (xxx.Frq). The input system is either continuous system (s-plane) described by a set of four state-space matrices or a discrete system (z-plane) described by four state-difference matrices. In addition to the quadruple matrices, the program also reads the sampling period (δt), and the system title. If the system is continuous ($\delta t=0$). Some of the program options are initialized from a settings table that includes some default values. For example, two algorithms are available for calculating the system’s frequency response. The user may adjust the initialization settings before proceeding with the frequency response calculation. Other parameters to adjust are: the frequency range, the number of points etc. There is also a variable frequency step (VFS) option for generating smooth Nichol's and Nyquist plots. This option is important for measuring phase and gain margins in systems that have low damped resonances, such as structural or slosh modes. The program saves the frequency response data in file xxx.Frq and calls the graphics processor to plot the frequency data on the screen.

Graphics Post-Processor: The graphics processing program reads the frequency response data from one or two (.Frq) files and plots it on the screen in three different forms. From the options menu, the user may choose to display the data in Bode, Nichol's, or Nyquist plots. The post-processing program either plots one curve from one file, or it can overlay two frequency response curves on the same plot from two separate frequency response files. The first curve from the first file, x1.frq, appears in blue. The second overlay curve from a second file x2.frq appears in red. The two sets of data from the two files must be compatible with each other for overlay in terms of frequency range. The graphics post-processor allows the user to display the frequency response data in the user desired form. The user may point the cursor on the locus and read the gain, phase, and frequency at that point. With the mouse you may also focus in a smaller area of the plot and magnify that area in order to observe more details there.

The graphics processor is also used to analyze stability of non-linear systems and evaluate the existence and size of limit-cycles by using the Describing Function (DF) method. The system’s frequency response is calculated across the non-linearity and the inverse of the DF is plotted in the same Nichols or Nyquist diagram. The user must provide the DF of the non-linearity in a separate file that has an extension (.DF). This file contains: the gain and phase of the fundamental non-linearity output calculated at different sinusoidal input amplitudes. The DF is either obtained analytically or experimentally using Simulink models. See examples for details.

Program Files

The frequency response analysis program uses three different types of files:

Standard Flixan systems file (.Qdr): This file contains the state-space systems to be analyzed. The systems are either continuous or discrete. The program only reads data from this file. The systems are generated either from a modelling program, transfer functions interconnections, or by the systems interconnection program. The systems file is associated with a certain vehicle analysis and it may contain several systems to be analyzed. The user must select one system at a time to be analyzed and specify the input/ output path. When the program completes the frequency response calculation of a system, another input/ output path or another system can be selected for analysis from the same file, and the process is repeated.

Frequency Response File (.Frg): The program generates a frequency response data file that has an extension (.Frg). This file is also used as input to the graphics post-processor. The first part of the filename is identical to the systems file. Only the extension is different. The graphics program can accept one or two frequency data files, but when two files are used the data must be compatible for overlay (same frequency range and number of points). Each frequency file may contain more than one set of frequency response data that are generated in a single process from one systems file (.Qdr).

The first line of each frequency data-set includes the number of the system's inputs and outputs, and also some parameters which are passed to the post-processing program. The second line above the frequency data includes the system title, which is the same as the title that appears in the systems file above the system quadruples. The input and output numbers across which the system's frequency response was calculated, including short input/ output definitions are also included below the title, and the number of frequency points.

The frequency response file consists of five columns of data. The first column contains the frequencies in (rad/sec), the second and third columns consist of the real and the imaginary parts of the transfer function, the third column includes the Gain in (dB), and the fifth column is the system's phase in (degrees). The first three characters on the left side of the frequency file contain reference numbers used by the graphics post-processor to locate and read the appropriate data, and also to navigate forwards and backwards in the frequency data file.

```
1 Number of Inputs= 6      Outputs= 14      M-Circle, Gain and Phase Margins:  2.00      8.00      40.0
  Plant Model, Vehicle/Actuators/Sensors (Z-Transform T=0.002)
2 Frequency Response for the following Transfer Function path
  Output( 2)-Roll Rate to FCS      / Input( 1)-Roll FCS Command (DP-TVC) ; Decades= 5
5
  NPT= 8000 OMEGA      X      Y      GAIN(DB)      PHASE(DEGR)
  1  0.100000E-02  -0.701176E+00  -0.222287E+01  0.735022E+01  -0.107507E+03
  2  0.100340E-02  -0.705218E+00  -0.222950E+01  0.737828E+01  -0.107553E+03
  3  0.101358E-02  -0.717412E+00  -0.224932E+01  0.746175E+01  -0.107690E+03
```

The Describing Function File (.DF): This file is only included when analyzing stability of non-linear systems using the DF method. It contains the title of the non-linearity followed by the DF data. That is: in the first column we have the amplitude of the sinusoidal that is applied to the non-linearity, and in the second and third columns we have the gain and phase of the fundamental frequency coming out of the non-linearity for the corresponding input amplitude. The DF is either obtained analytically or experimentally using Simulink models.

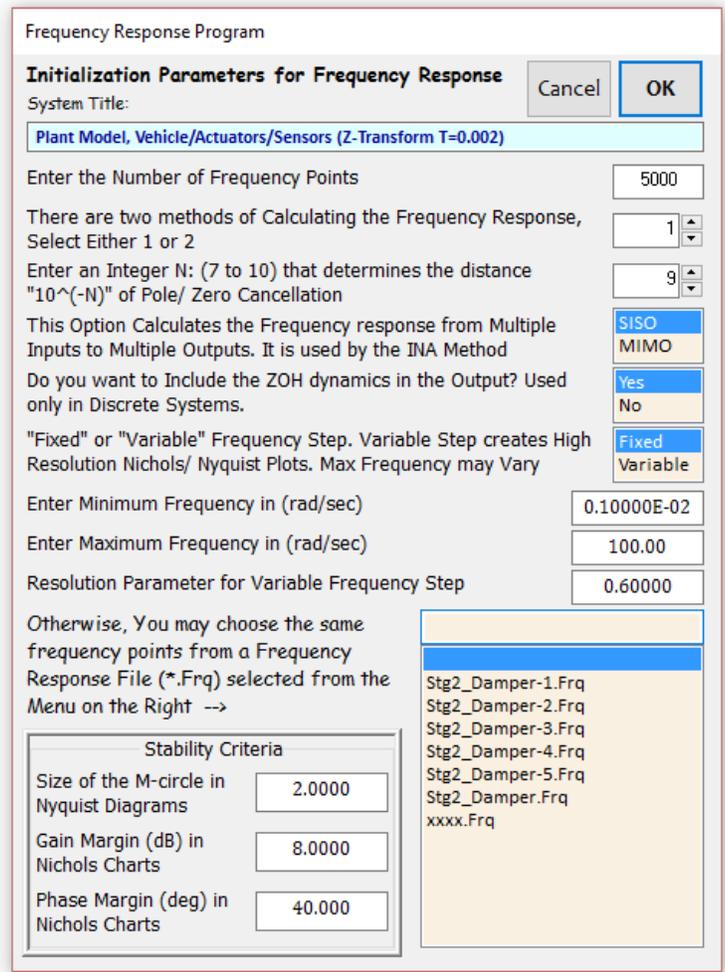
User defined Non-Linearity

Amplitude	Gain	Phase (deg)
0.1000	0.5000	0.0000
0.1250	0.5300	0.0000
0.1750	0.6000	0.0000
0.2000	0.6300	0.0000
0.2250	0.6700	0.0000
0.2750	0.7500	-0.5000
0.3000	0.8000	-1.0000
0.3200	0.7800	-2.0000

Program Options

After selecting the systems file and then one of the systems to analyze, the program requires some options to be selected by the user and the defaults are shown in the following menu.

1. The number of points to be included in the frequency response calculation, it should be less than 20,000 points.
2. The program provides two methods for calculating the system's frequency response. Method #1 is significantly faster than method #2. We recommend using the default method #1 because it is faster. Use method # 2 as a backup.
3. The next option defines the accuracy of the frequency response calculations that you want to obtain. The program computes the poles and zeros of the system between a specified input and a specific output. When the poles and zeros are very close together, their influence on the frequency response is negligible and canceling them out simplifies the calculations. The amount of cancellation between poles and zeros depends on a number (n) that must be entered. This number must be in the range 7-10, depending on the amount of cancellation desired. The number (n) defines the cancellation distance d between poles and zeros, ($d=10^{-n}$). If there is a pole/ zero pair that are closer together than d, they will cancel out



and they will not affect the calculations. If the transfer gain between a system's input and output is very small, we recommend that you increase the value of n to 12 or 14. However, if the system to be analyzed has a very low gain or very high gain, i.e. the magnitudes of the elements of matrices B, C, D are very small (in the order of 10^{-9} or less), or very high (10^{+9} or more), we recommend that you scale this system up or down accordingly before analyzing it. This is a good practice in almost every application.

4. The next option is used for calculating a multivariable (MIMO) frequency response. That is, from multiple inputs to multiple outputs. The default selection is SISO, that is, one input/ output pair at a time. The only time that you would choose the MIMO option is when you want to allow the program to automatically compute frequency responses between multiple inputs and multiple outputs in one batch, that is, from every input to every output. This is useful when performing multivariable frequency response analysis such as "Inverse Nyquist Arrays". When you select the multivariable option the program will ask you to select the input numbers and the output numbers of the system that you want to compute the MIMO frequency response. Note that in order to apply the INA method the number of inputs must be equal to the number of outputs, so the program assumes that they are equal.
5. The next initialization option is applies only in discrete-time systems and it has to do with the option of including or not the dynamic effect of the zero-order-hold in the frequency response computations. The zero-order-hold will introduce some additional phase delay and attenuation especially at frequencies near the Nyquist frequency. The default zero-order-hold value is "include" and it must be included when performing open-loop calculations for stability analysis.
6. The user must also specify the frequency range by entering the initial and final frequencies.
7. The next option is for defining the frequency step size in the frequency response computation. When the selection is "Fixed", which is the default, the frequency step will be defined from the number of points and the specified frequency range. If the selection is "Variable", a variable frequency step (VFS) is selected. This option is useful for generating smooth and nice looking Nichols and Nyquist curves by spreading the points evenly to maintain an nearly constant resolution. The VFS option allows the frequency step between computations to vary according to gain and phase variation. When the gain and phase do not change significantly with frequency, such as at very low frequencies, the frequency step is increased. When the gain and phase begin to change more rapidly with frequency, such as at low damped resonances, the frequency step is reduced. The relative step size is adjusted by the resolution parameter that must be entered by the user. The default value is 0.6. The disadvantage of using the VFS option is that it may require several attempts to cover the specified frequency range by adjusting the resolution parameter.
8. There is a third option for defining the frequency points in the frequency calculation and that is to use the same frequency points as those defined from a previously calculated (.Frq) file. This is useful when overlaying frequency data obtained from similar systems and you want them to be in the same frequency range and include the same number of points at the same frequencies, especially when the first file is calculated using the VFS option. In order to activate this option you must select the (.Frq) file from the menu shown in the initialization menu above.

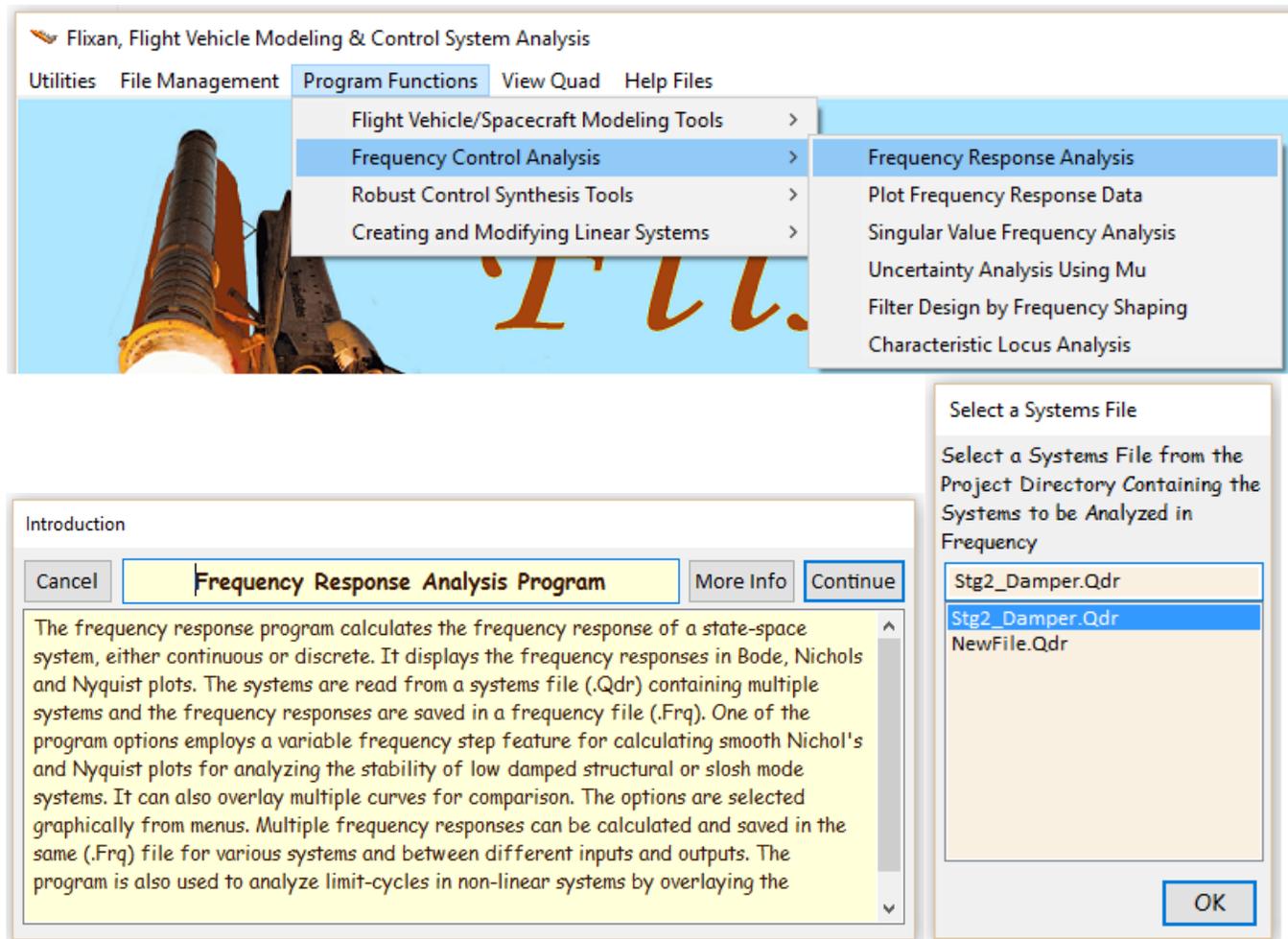
9. Finally, the initialization parameters include the user defined gain and phase margins that are used for plotting the area to avoid rectangle around the critical point. The Nichols locus must avoid this rectangle in order to satisfy the required gain and phase margins. The size of the M-circle is also defined which plays a similar role in the Nyquist diagrams.

Running the Frequency Response Program

We will demonstrate the frequency response analysis program using two examples. The first example calculates responses of several systems from a systems file, and the second example demonstrates the use of the variable frequency step and overlaying two frequency response files.

Example 1 Frequency Response Calculations from Multiple Systems

Our first example is in folder “C:\Flixan\Frequ\Examples\Ex1”. We will calculate and plot frequency responses of systems which are located in systems file “Stg2_Damper.Qdr”. Start the Flixan program, select the project directory and from the main menu select “Program Functions”, “Frequency Control Analysis”, and then “Frequency Response Analysis”. The following is an introduction dialog and click “Continue”. Select also the systems file from the next menu and click “OK”.



The systems file contains several systems. Using the systems selection menu below, select one of the system titles "Shuttle Main Engine Hydraulic Actuator (Type-I)" and click on "Select". Use the dialog below to set the initialization parameters for the actuator frequency response calculations, as shown, and click "OK".

Select a State-Space System from Quad File

Select a State-Space Model for Frequency Analysis From Systems File: Stg2_Damper.qdr

Shuttle Second Stage with Payload Damper, at T=123 sec
Shuttle Main Engine Hydraulic Actuator (Type-I)
 Actuators/TVC (Second Stage)
 Sensor Dynamics
 Plant Model, Vehicle/Actuators/Sensors
 Plant Model, Vehicle/Actuators/Sensors (Z-Transform T=0.04)
 Plant Model, Vehicle/Actuators/Sensors (Z-Transform T=0.002)
 Shuttle Stage-2 Continuous Flight Control System
 Shuttle Stage-2 Discrete Flight Control System
 Pitch Open-Loop Model (s-plane)
 Pitch Open-Loop Model (z-plane)

Choose a System Title and then click "Select" Cancel View System Select

Frequency Response Program

Initialization Parameters for Frequency Response Cancel OK

System Title:
Shuttle Main Engine Hydraulic Actuator (Type-I)

Enter the Number of Frequency Points

There are two methods of Calculating the Frequency Response, Select Either 1 or 2

Enter an Integer N: (7 to 10) that determines the distance "10^{-N}" of Pole/ Zero Cancellation

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method SISO MIMO

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems. Yes No

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary Fixed Variable

Enter Minimum Frequency in (rad/sec)

Enter Maximum Frequency in (rad/sec)

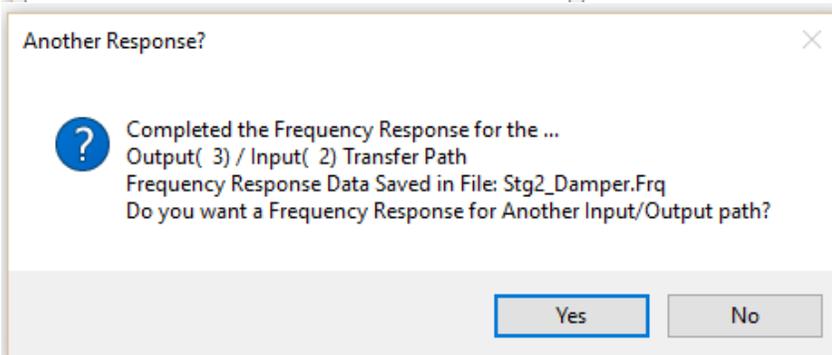
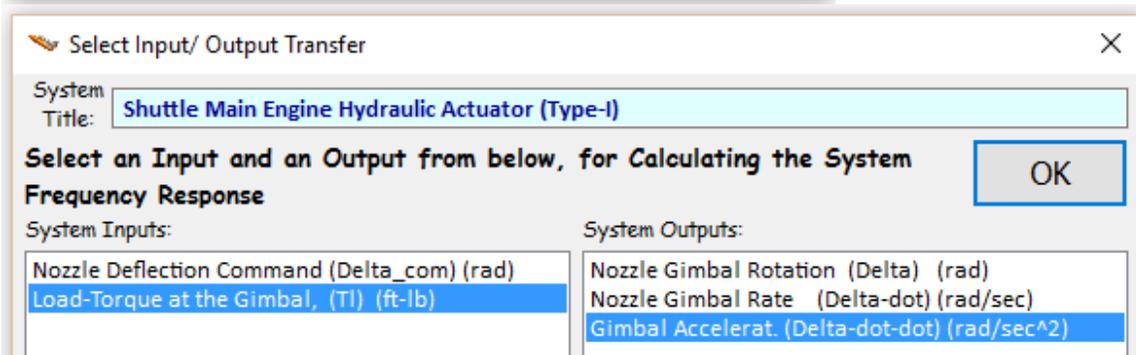
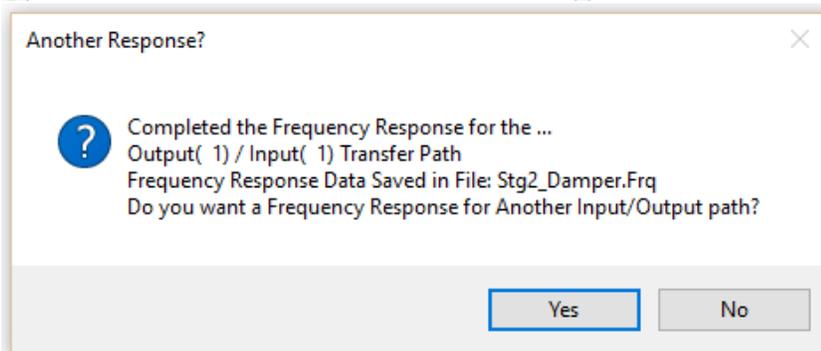
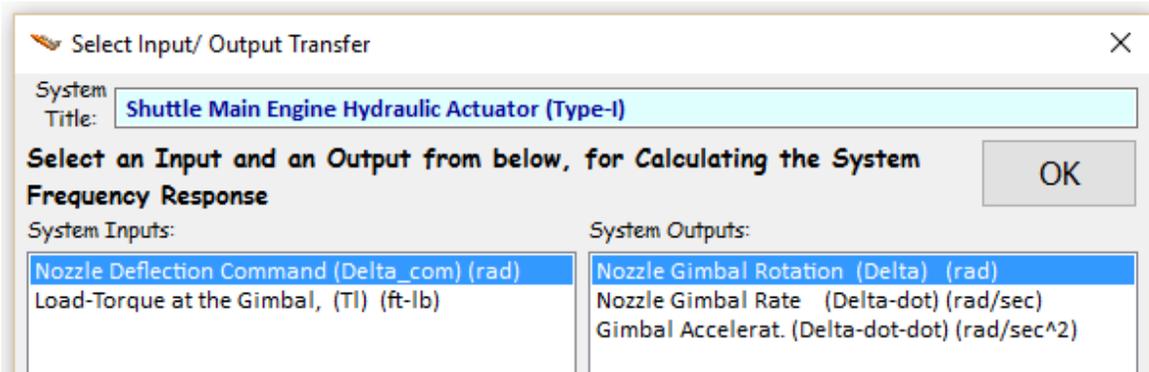
Resolution Parameter for Variable Frequency Step

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

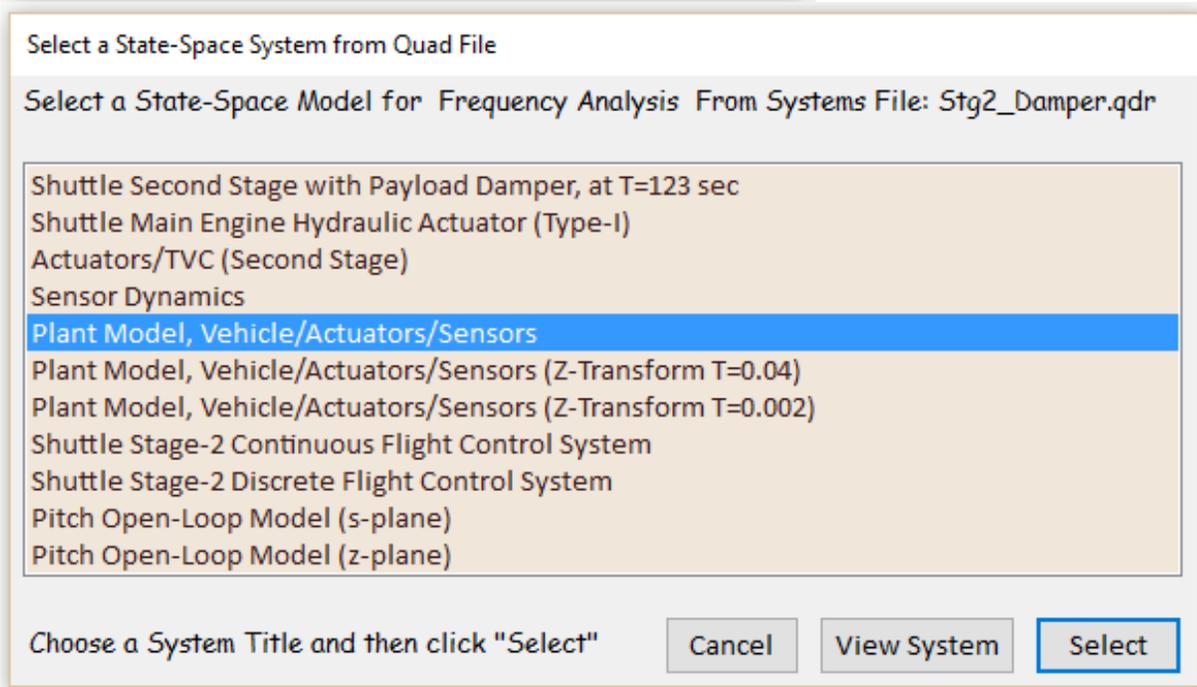
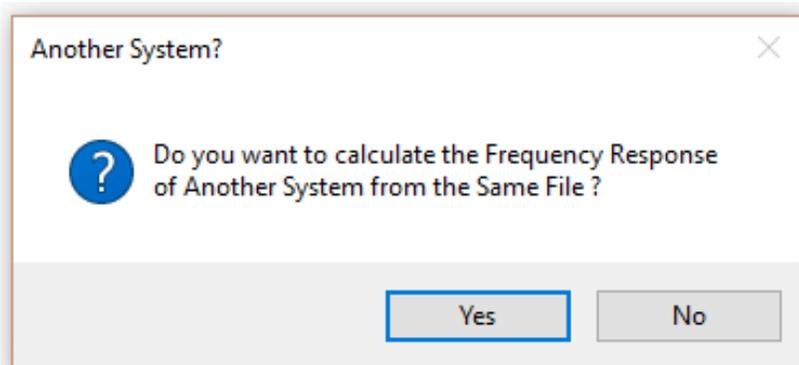
Stability Criteria	
Size of the M-circle in Nyquist Diagrams	<input type="text" value="2.0000"/>
Gain Margin (dB) in Nichols Charts	<input type="text" value="8.0000"/>
Phase Margin (deg) in Nichols Charts	<input type="text" value="40.000"/>

Stg2_Damper-1.Frq
 Stg2_Damper.Frq
 x1.Frq
 x2.Frq

You must now define the actuator system's input and output across which the program will calculate the frequency response. Select first the "Nozzle Command" input and the "Nozzle Deflection" output, and click "OK". The program calculates the frequency response for the selected input/output pair and asks the user if he wishes to calculate another frequency response for another input/output pair. Click on "Yes" and calculate the frequency response between the "Load-Torque" input and the "Nozzle Gimbal Acceleration" output, as shown.



When you finish, answer “No” that you don’t wish to calculate another frequency response using this actuator system. You would like, however, (answer “Yes” below) to calculate the frequency response of another system from the same file, and from the following menu select the system “*Plant Model, Vehicle/ Actuators/ Sensors*”, and click “*Select*”.



The following dialog is used to set the parameters in the calculation of the frequency response of the "Plant Model" system. We are using 10,000 points because this system has a lot of structural and slosh modes.

Frequency Response Program

Initialization Parameters for Frequency Response Cancel OK

System Title:

Enter the Number of Frequency Points

There are two methods of Calculating the Frequency Response, Select Either 1 or 2

Enter an Integer N: (7 to 10) that determines the distance "10^(-N)" of Pole/ Zero Cancellation

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method
 SISO
 MIMO

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems.
 Yes
 No

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary
 Fixed
 Variable

Enter Minimum Frequency in (rad/sec)

Enter Maximum Frequency in (rad/sec)

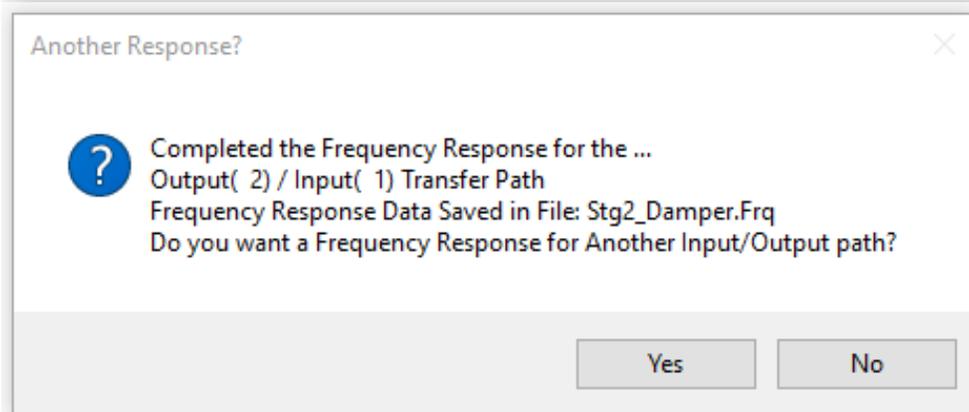
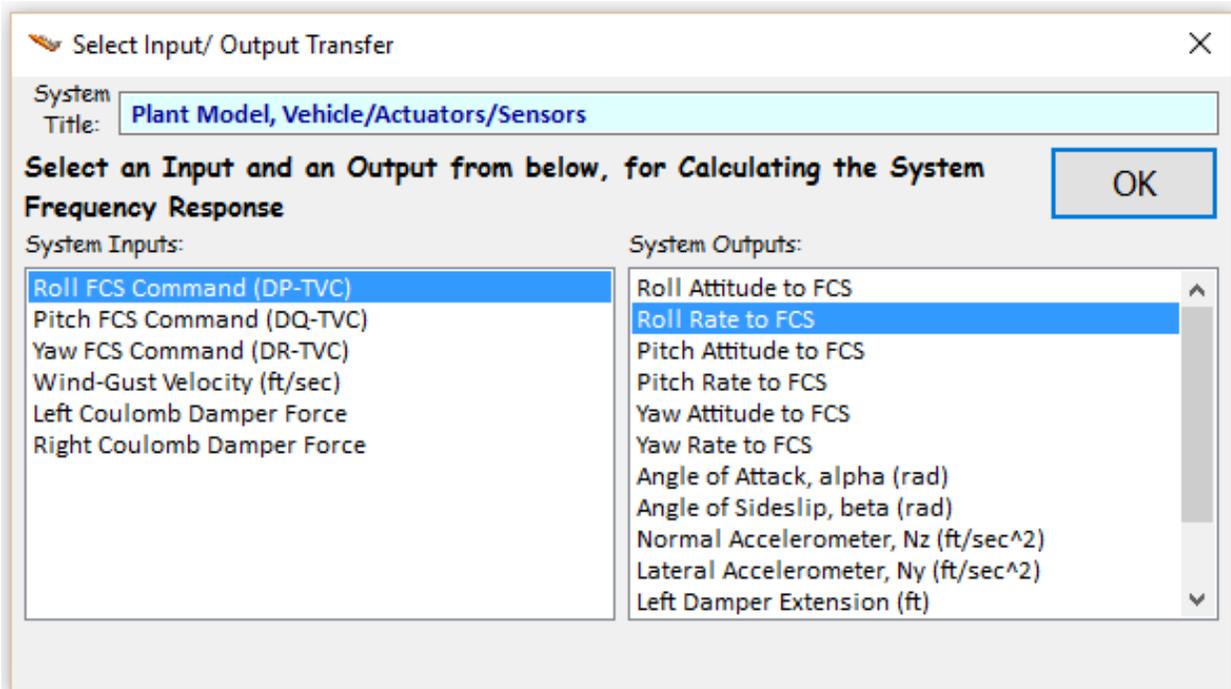
Resolution Parameter for Variable Frequency Step

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

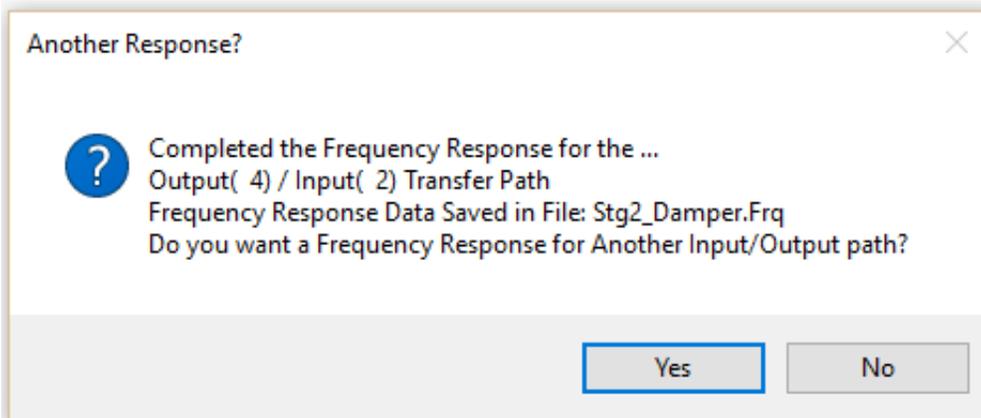
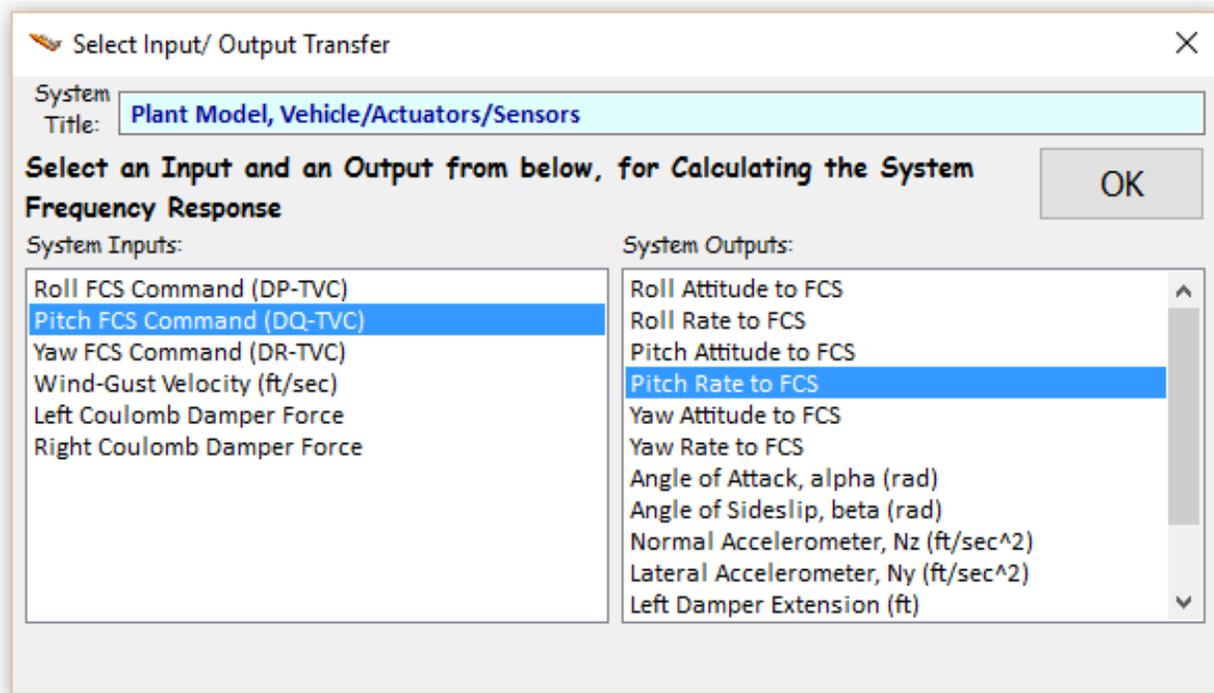
Stability Criteria	
Size of the M-circle in Nyquist Diagrams	<input type="text" value="2.0000"/>
Gain Margin (dB) in Nichols Charts	<input type="text" value="8.0000"/>
Phase Margin (deg) in Nichols Charts	<input type="text" value="40.000"/>

- Stg2_Damper-1.Frq
- Stg2_Damper.Frq
- x1.Frq
- x2.Frq

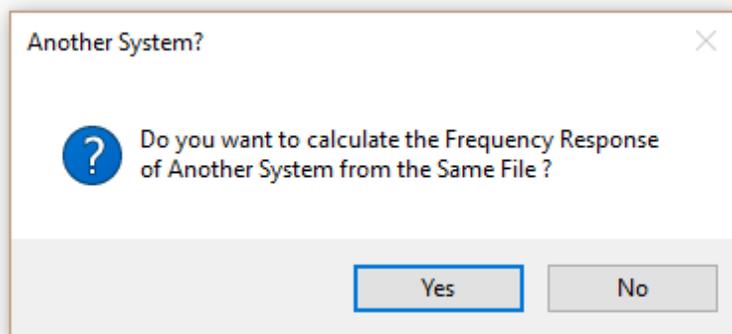
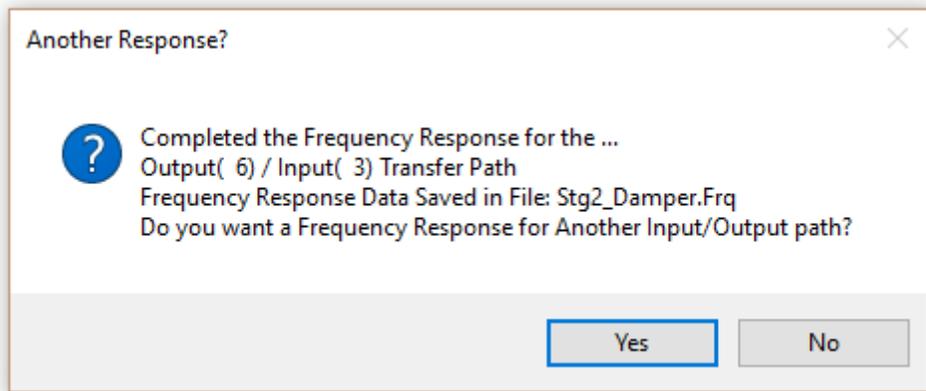
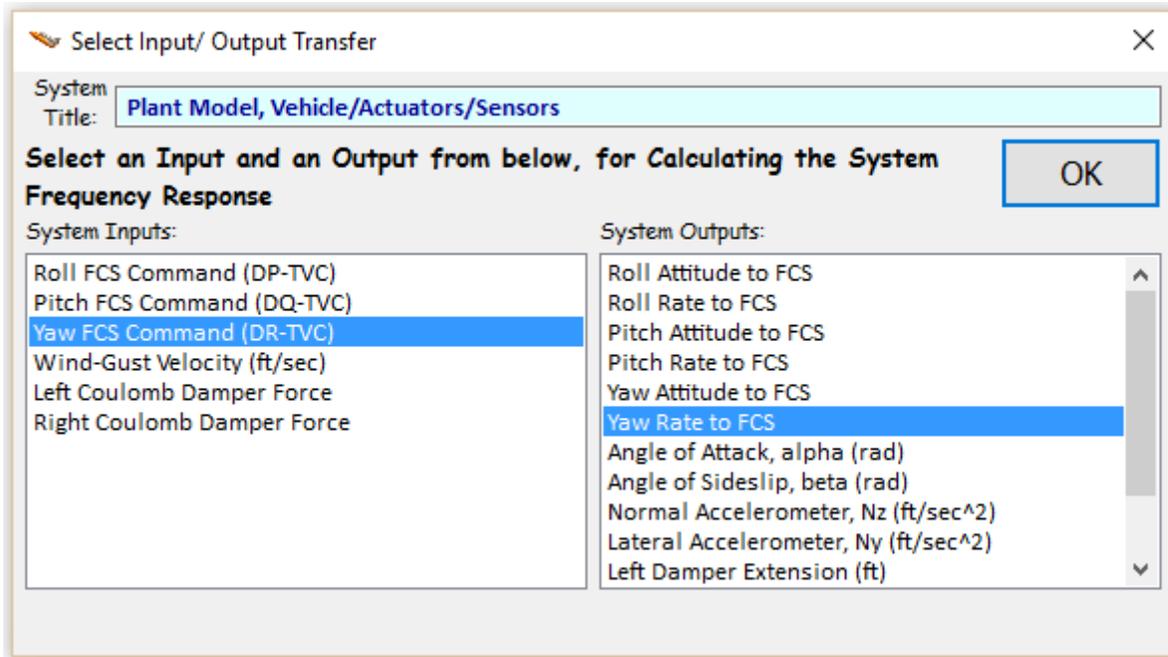
We will use this Plant Model system to calculate three frequency responses for the Roll, Pitch and Yaw axes, as shown below. In the menus below select the input and output for the roll axis. Click on “OK”, and in the next dialog click on “Yes” to select another input/output pair for the pitch axis.



In the next dialog select the input and output for the pitch axis. Click on “OK”, and in the next dialog click on “Yes” to select another input/output pair for the yaw axis.



Select an input/output pair for the yaw axis and click on “No” that you don’t want to compute another frequency response using the Plant Model system. In the next dialog answer “Yes” to select another system for frequency response analysis. This time select “Shuttle Stage-2 Continuous Flight Control System” and use the next dialog to set the program parameters as before.



Select a State-Space System from Quad File

Select a State-Space Model for Frequency Analysis From Systems File: Stg2_Damper.qdr

- Shuttle Second Stage with Payload Damper, at T=123 sec
- Shuttle Main Engine Hydraulic Actuator (Type-I)
- Actuators/TVC (Second Stage)
- Sensor Dynamics
- Plant Model, Vehicle/Actuators/Sensors
- Plant Model, Vehicle/Actuators/Sensors (Z-Transform T=0.04)
- Plant Model, Vehicle/Actuators/Sensors (Z-Transform T=0.002)
- Shuttle Stage-2 Continuous Flight Control System**
- Shuttle Stage-2 Discrete Flight Control System
- Pitch Open-Loop Model (s-plane)
- Pitch Open-Loop Model (z-plane)

Choose a System Title and then click "Select"

Cancel

View System

Select

Frequency Response Program

Initialization Parameters for Frequency Response

Cancel

OK

System Title:

Shuttle Stage-2 Continuous Flight Control System

Enter the Number of Frequency Points

10000

There are two methods of Calculating the Frequency Response, Select Either 1 or 2

1

Enter an Integer N: (7 to 10) that determines the distance "10^{-N}" of Pole/ Zero Cancellation

9

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method

SISO

MIMO

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems.

Yes

No

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary

Fixed

Variable

Enter Minimum Frequency in (rad/sec)

0.10000

Enter Maximum Frequency in (rad/sec)

100.00

Resolution Parameter for Variable Frequency Step

0.60000

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

Stability Criteria

Size of the M-circle in Nyquist Diagrams

2.0000

Gain Margin (dB) in Nichols Charts

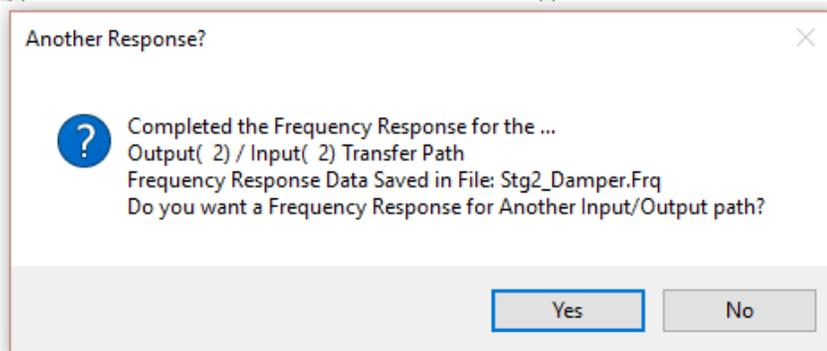
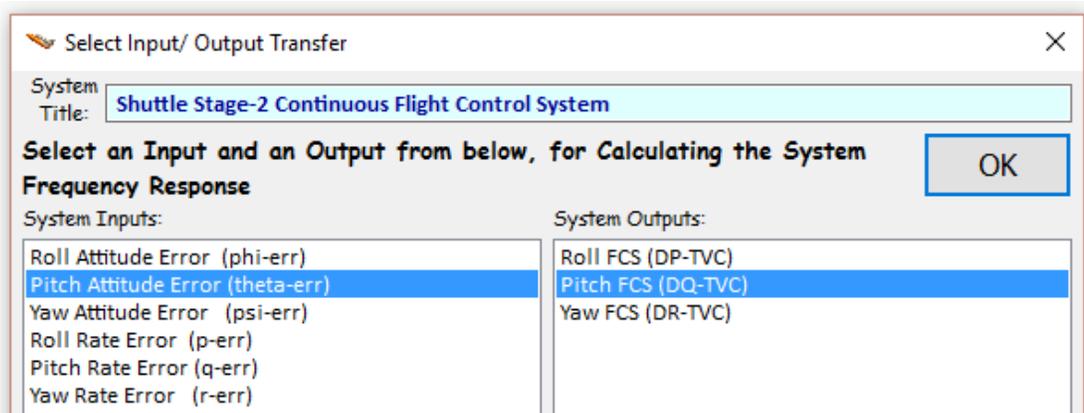
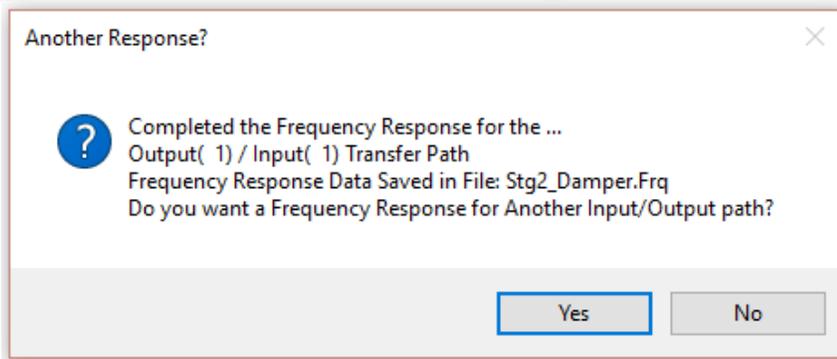
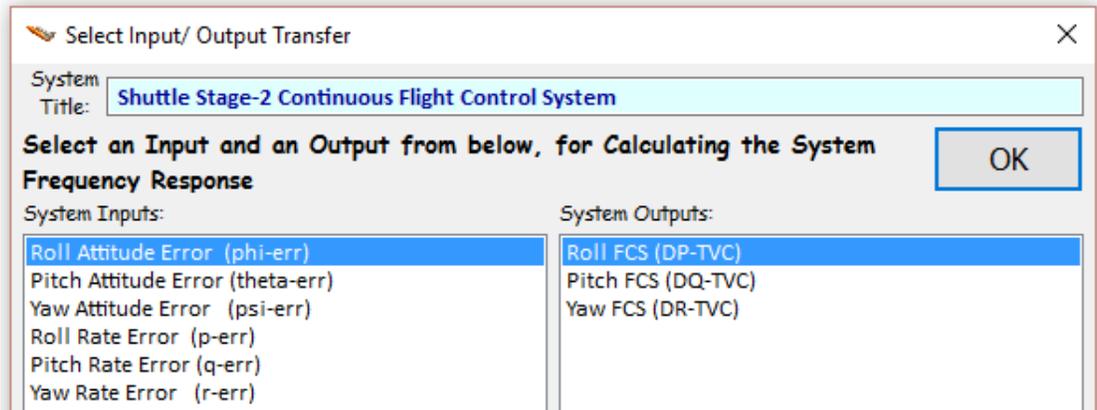
8.0000

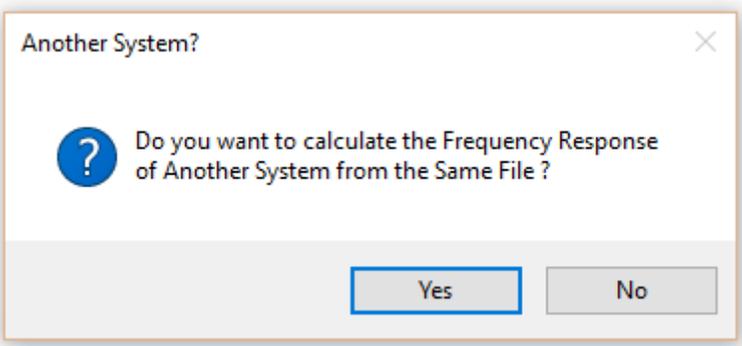
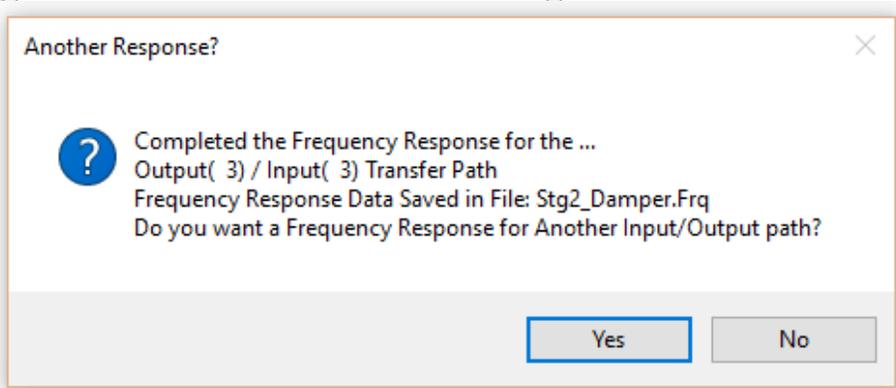
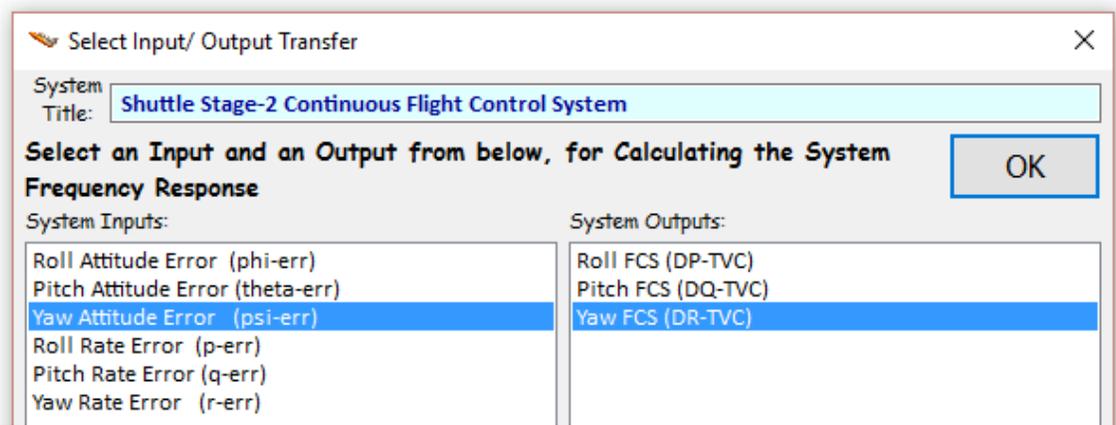
Phase Margin (deg) in Nichols Charts

40.000

a.Frq
Stg2_Damper-1.Frq
Stg2_Damper-2.Frq
Stg2_Damper.Frq
x1.Frq
x2.Frq

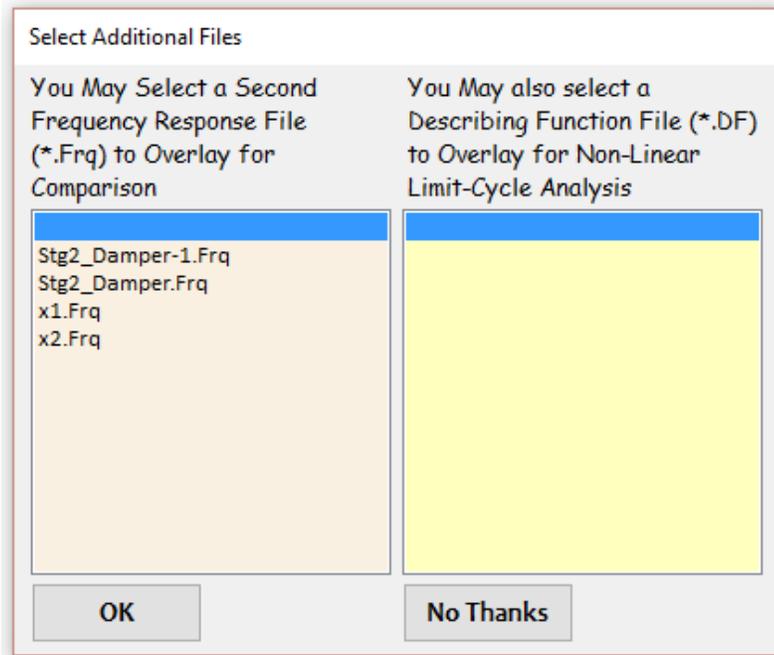
We will now analyze the discrete Shuttle flight control system and calculate the three frequency responses between roll, pitch and yaw attitude errors and the flight control system outputs.



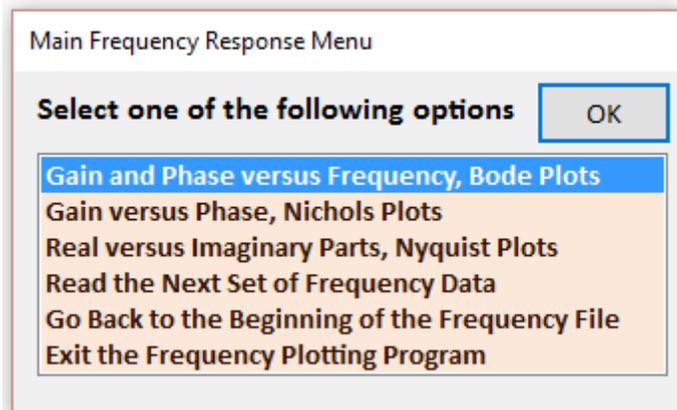


Click on “No” in both of the above dialogs and the program will plot the frequency response data which are saved in file “Stg2_Damper.Frq”.

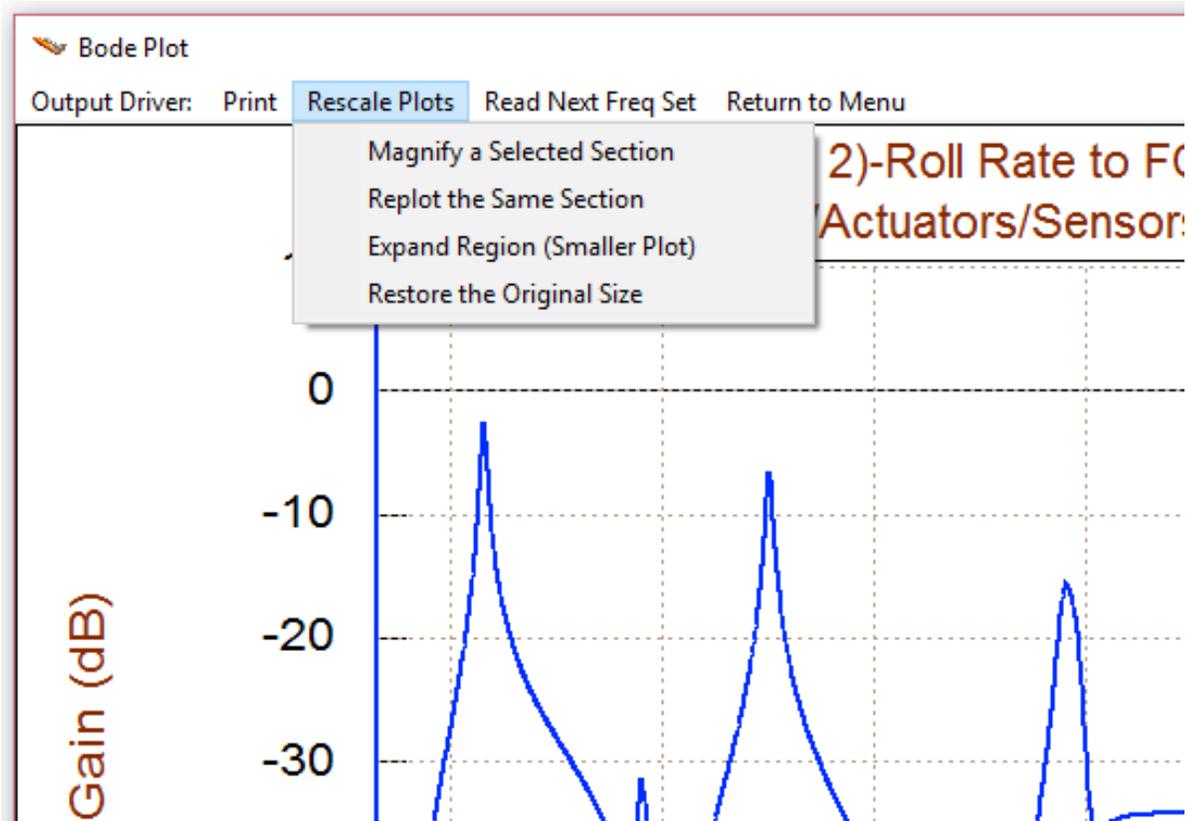
Click "OK" in the menus below without selecting any files because you don't want to overlay any data from another previously calculated (.Frq) file or from a "Describing Function" file.



The following menu is used for plotting the frequency response data in Bode, Nichols, and Nyquist formats. The frequency response data is read from file "Stg2_Damper.Frq" which contains multiple sets of frequency data from various systems. The user can advance to the next set or go back to previous sets.



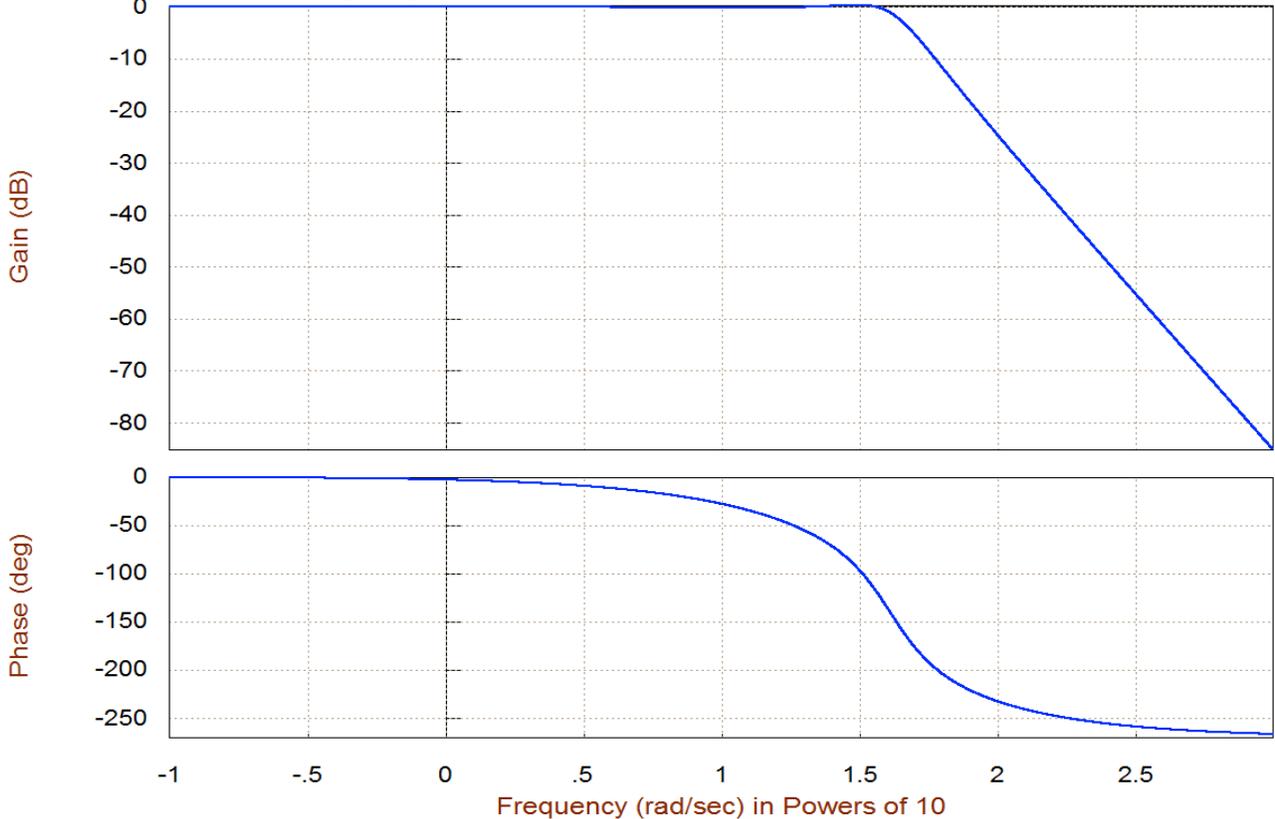
There is a menu located above each plot that is used to perform several functions. The first two options are used for selecting drivers for different output formats or for sending the plot to the printer. The “Rescale Plots” option has a drop-down menu used for adjusting the size of the plots, focusing in smaller areas or expanding the scales size to cover a larger area. You may also advance and plot the next set of frequency data or return to the main menu to choose a different plotting option.



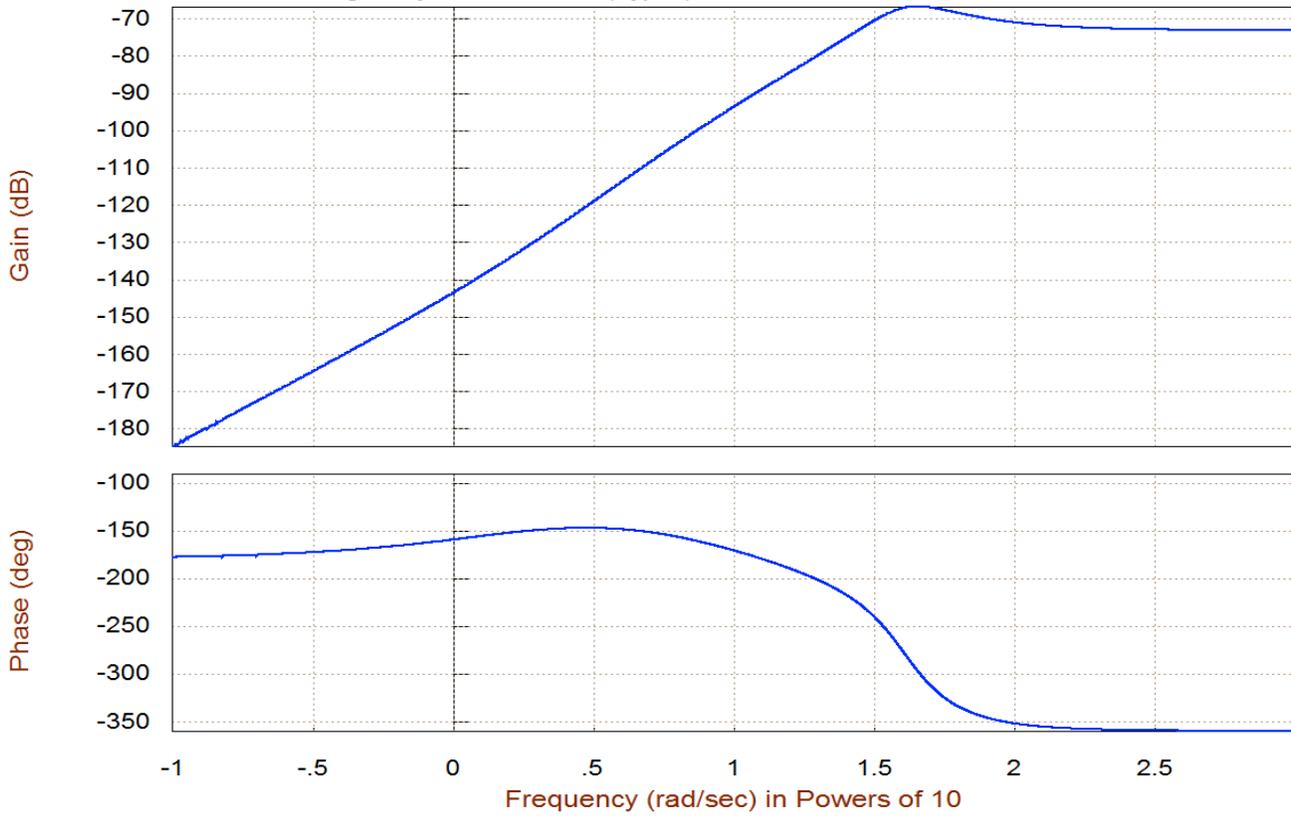
The following plots were obtained from the frequency response calculations described, beginning with two frequency responses from the actuator system, three responses from the Plant Model, and three responses from the Flight Control System. In separate plots below the regular Bode plots of the Plant Model, the regions around the flex modes are also shown expanded and in a linear scale. Those plots were obtained using the “Magnify a Selected Region” option from the menu. The peaks of the resonances are also labeled by clicking with the cursor on the actual locus.

The last three sets of plots show the frequency responses of the flight control system during second stage for the roll, pitch, and yaw axes calculated between the attitude error inputs and the control system outputs. They are presented in Bode and Nyquist plot formats.

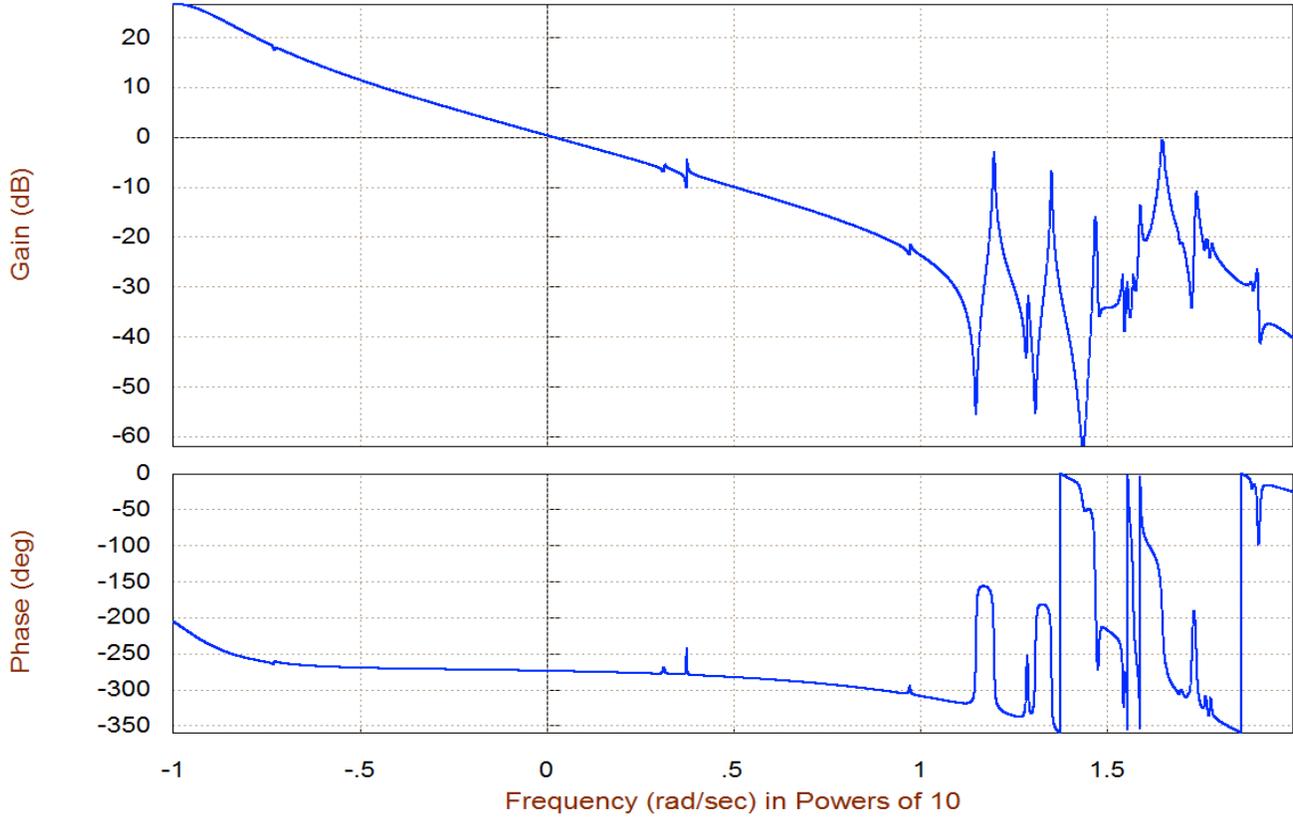
Bode Plot for: Outp(1)-Nozzle Gimbal Rotation (Delta / Inpt(1)-Nozzle Deflection Command (Del , of Shuttle Main Engine Hydraulic Actuator (Type-I)



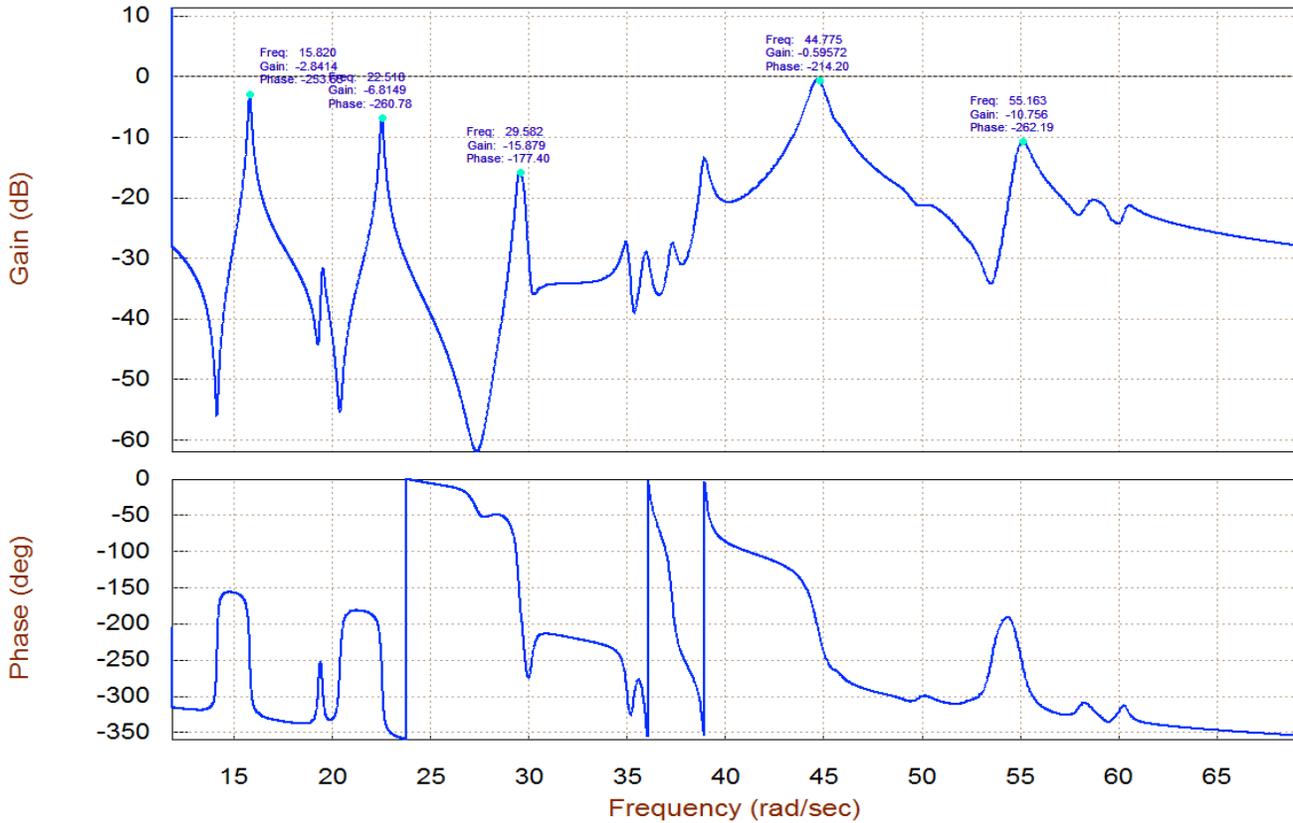
Bode Plot for: Outp(3)-Gimbal Accelerat. (Delta-dot-d / Inpt(2)-Load-Torque at the Gimbal, (T , of Shuttle Main Engine Hydraulic Actuator (Type-I)



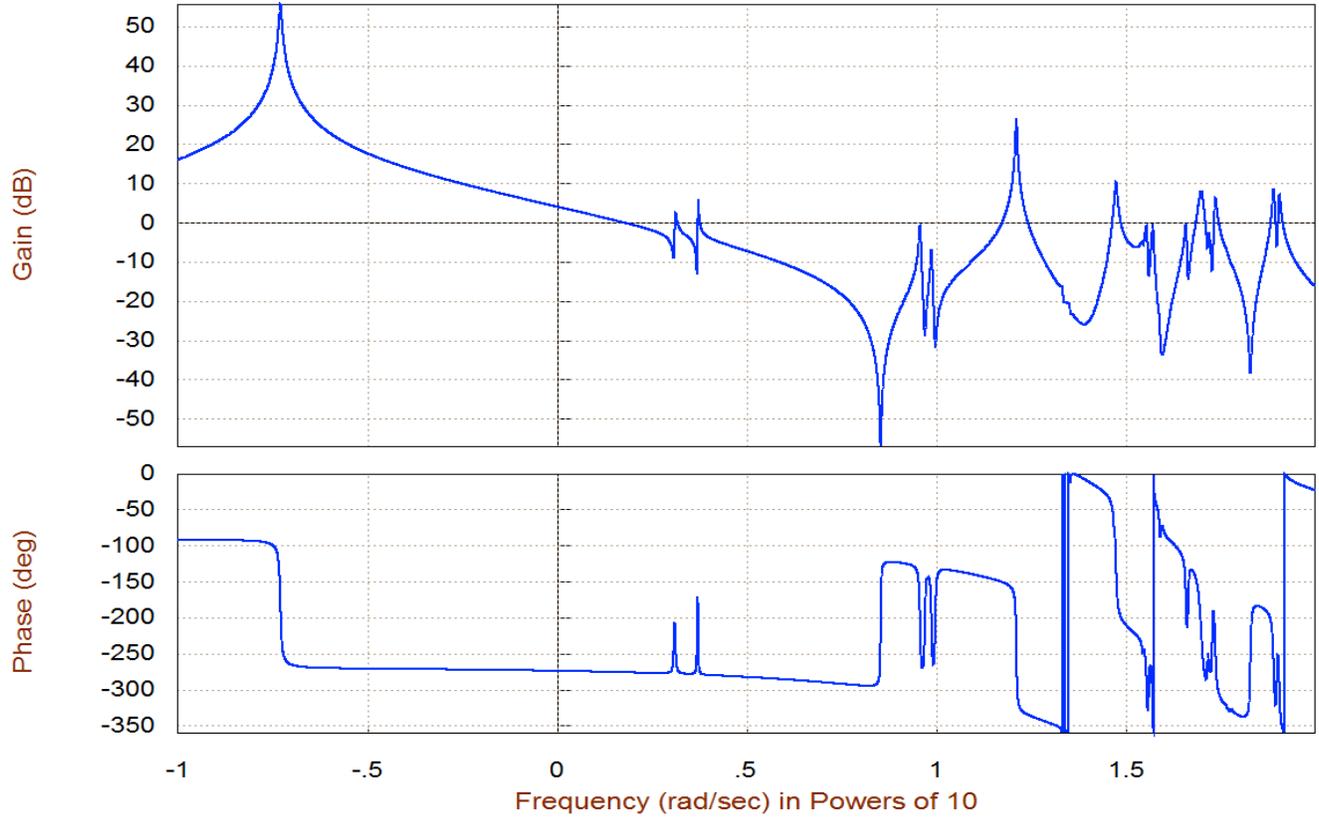
Bode Plot for: Outp(2)-Roll Rate to FCS / Inpt(1)-Roll FCS Command (DP-TVC) , of: Plant Model, Vehicle/Actuators/Sensors



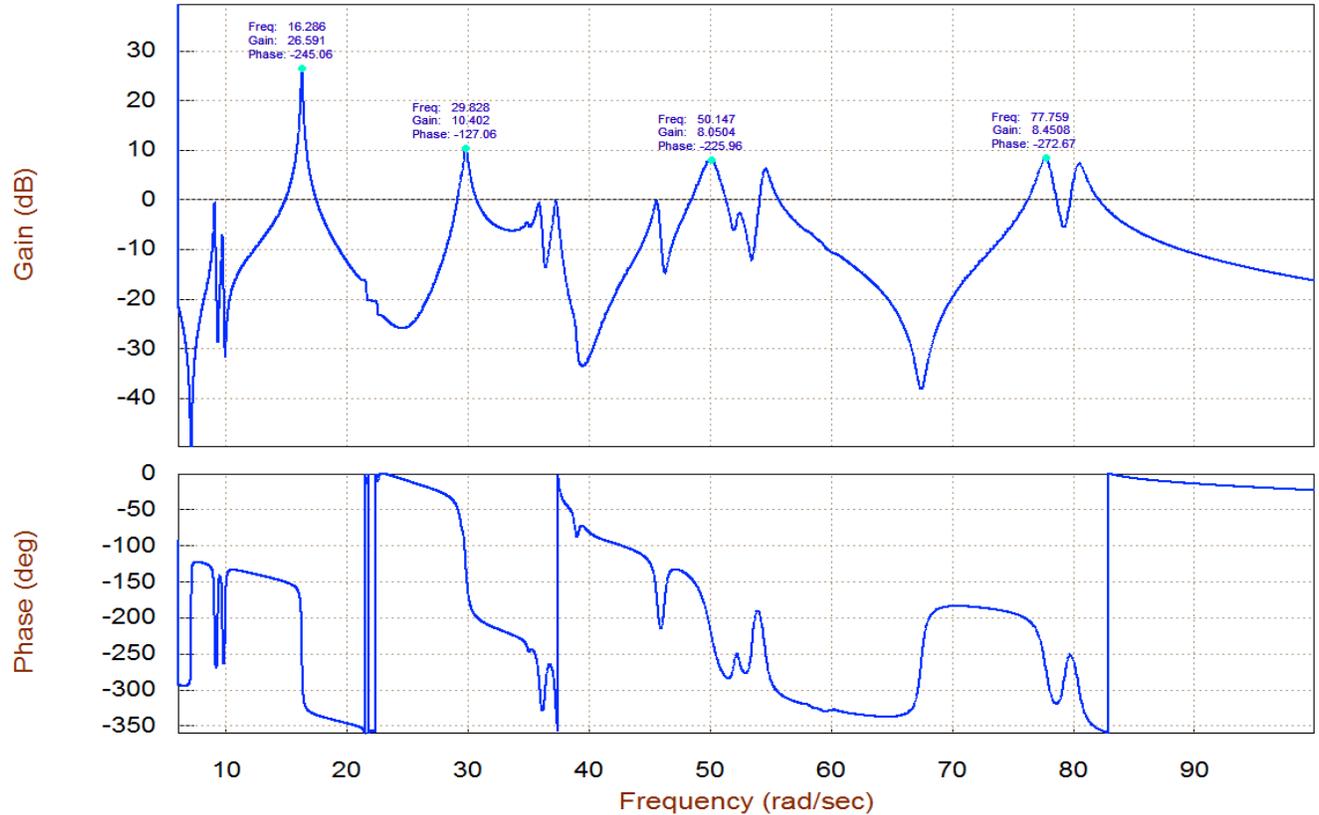
Bode Plot for: Outp(2)-Roll Rate to FCS / Inpt(1)-Roll FCS Command (DP-TVC) , of: Plant Model, Vehicle/Actuators/Sensors



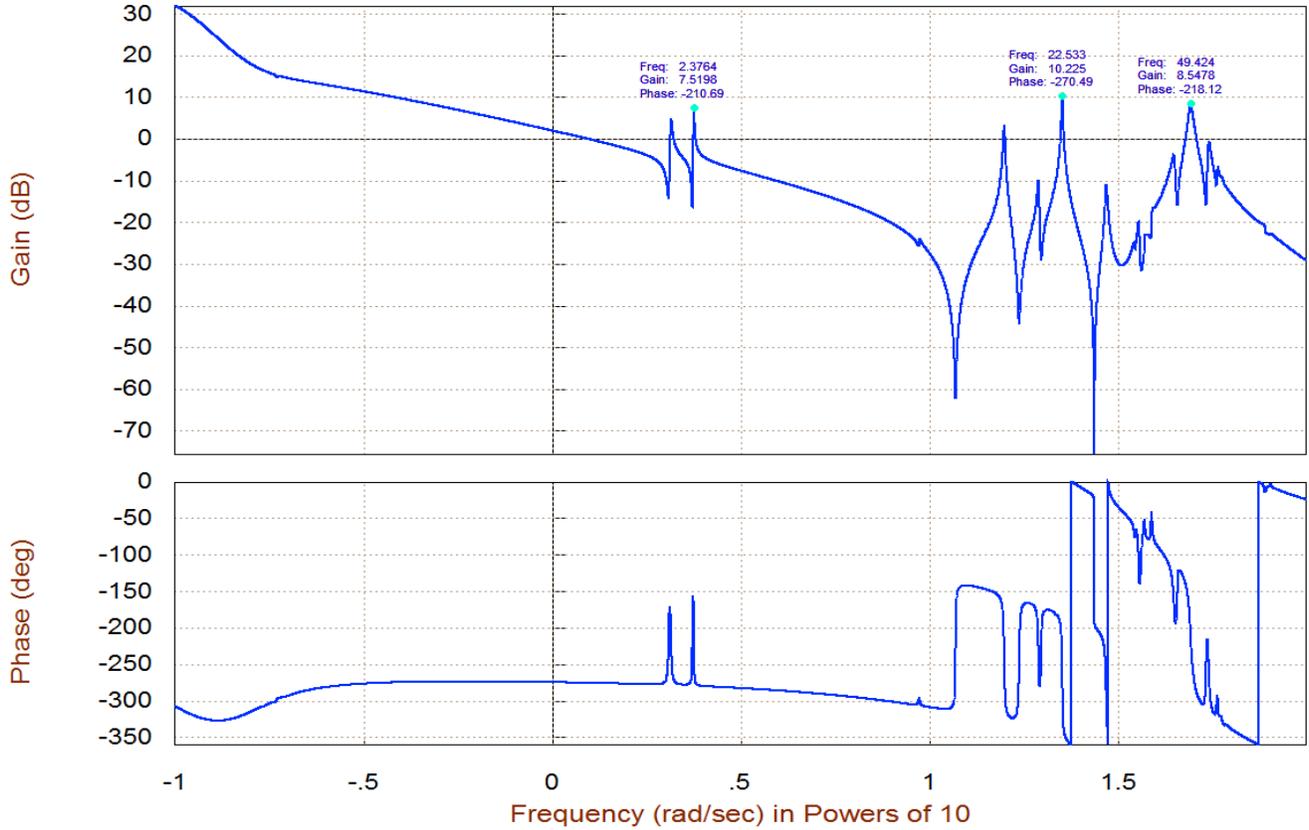
Bode Plot for: Outp(4)-Pitch Rate to FCS / Inpt(2)-Pitch FCS Command (DQ-TVC) , of: Plant Model, Vehicle/Actuators/Sensors



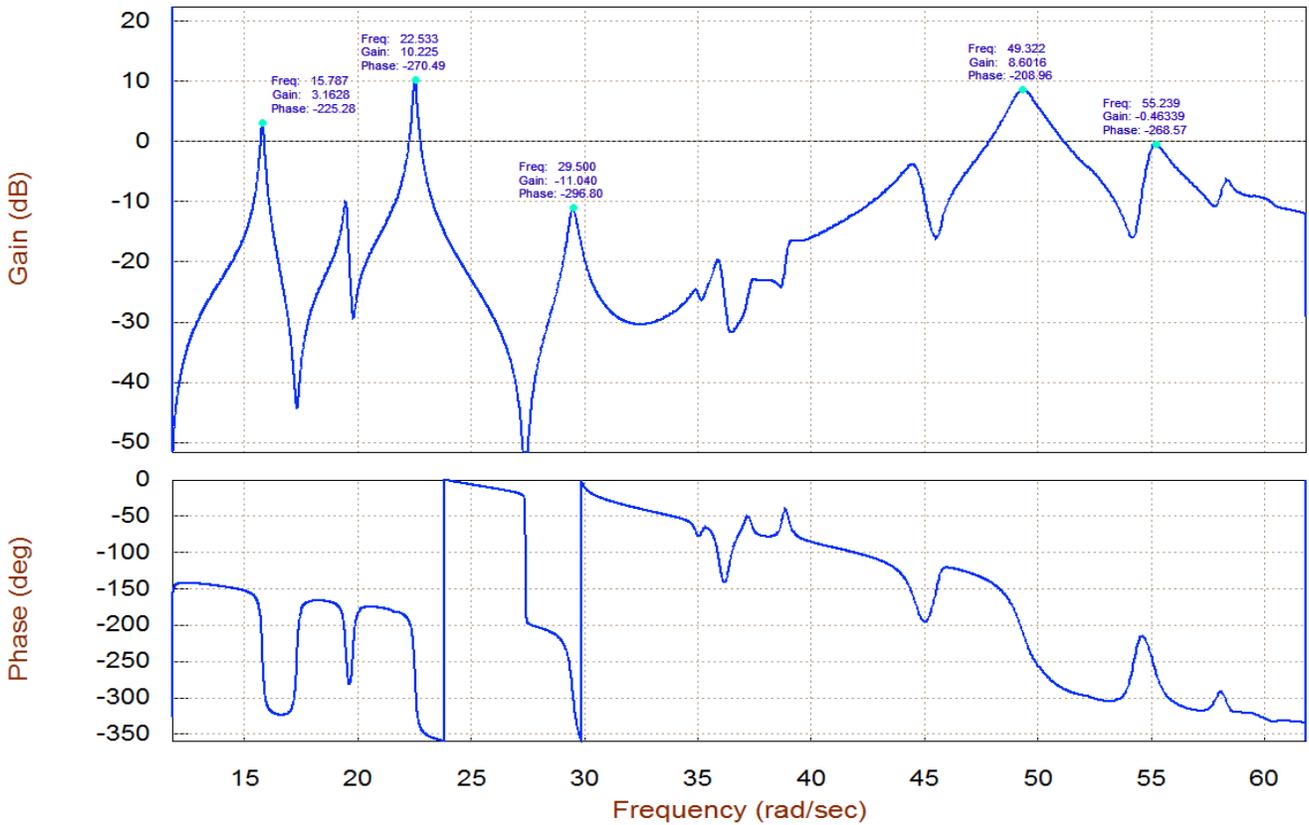
Bode Plot for: Outp(4)-Pitch Rate to FCS / Inpt(2)-Pitch FCS Command (DQ-TVC) , of: Plant Model, Vehicle/Actuators/Sensors



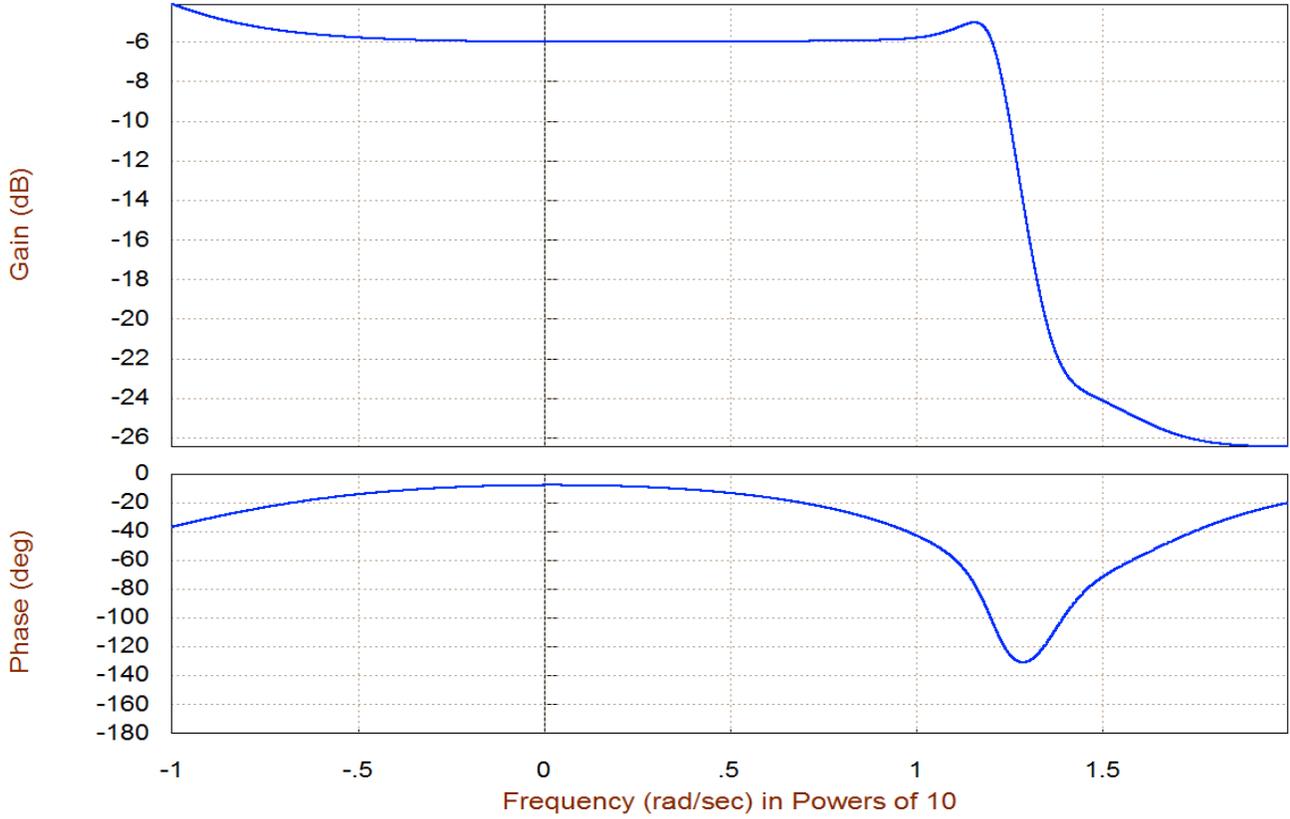
Bode Plot for: Outp(6)-Yaw Rate to FCS / Inpt(3)-Yaw FCS Command (DR-TVC) , of:
Plant Model, Vehicle/Actuators/Sensors



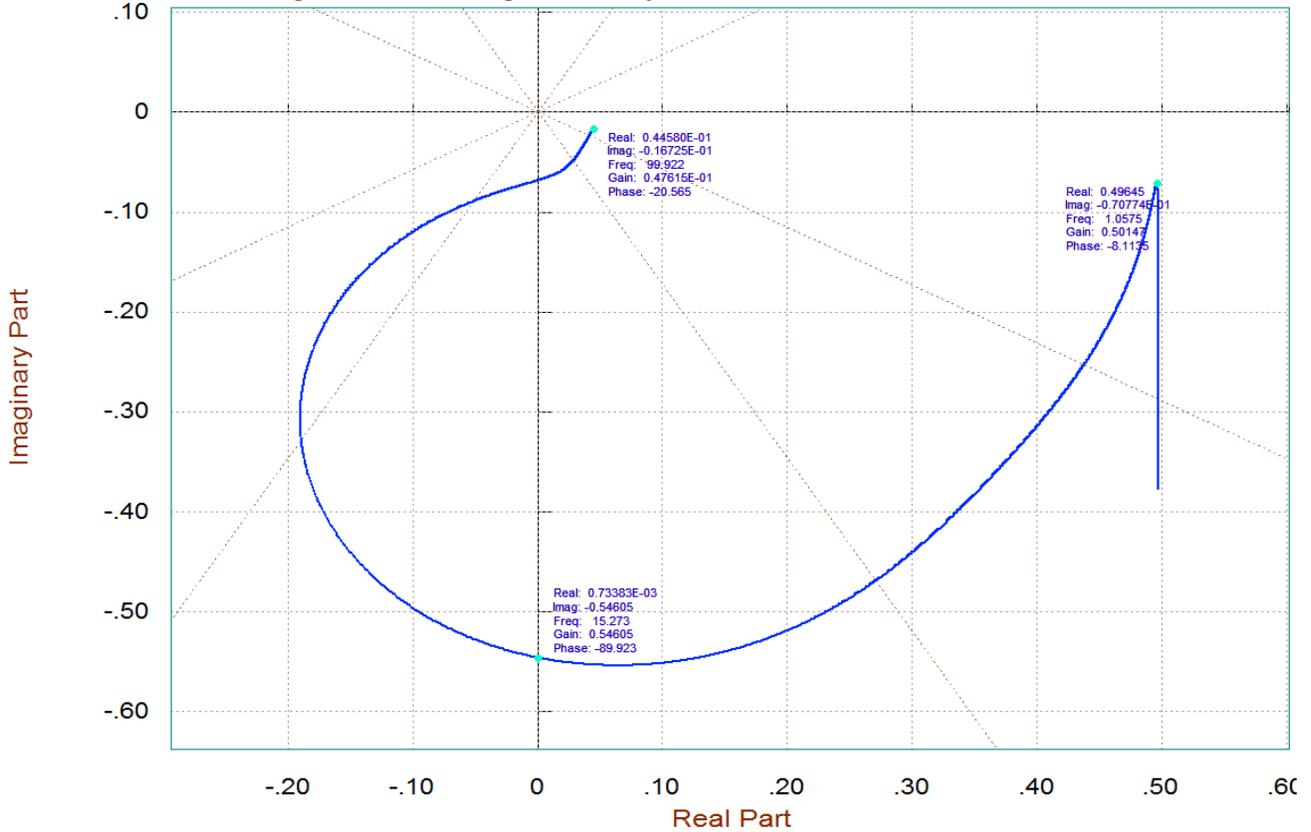
Bode Plot for: Outp(6)-Yaw Rate to FCS / Inpt(3)-Yaw FCS Command (DR-TVC) , of:
Plant Model, Vehicle/Actuators/Sensors



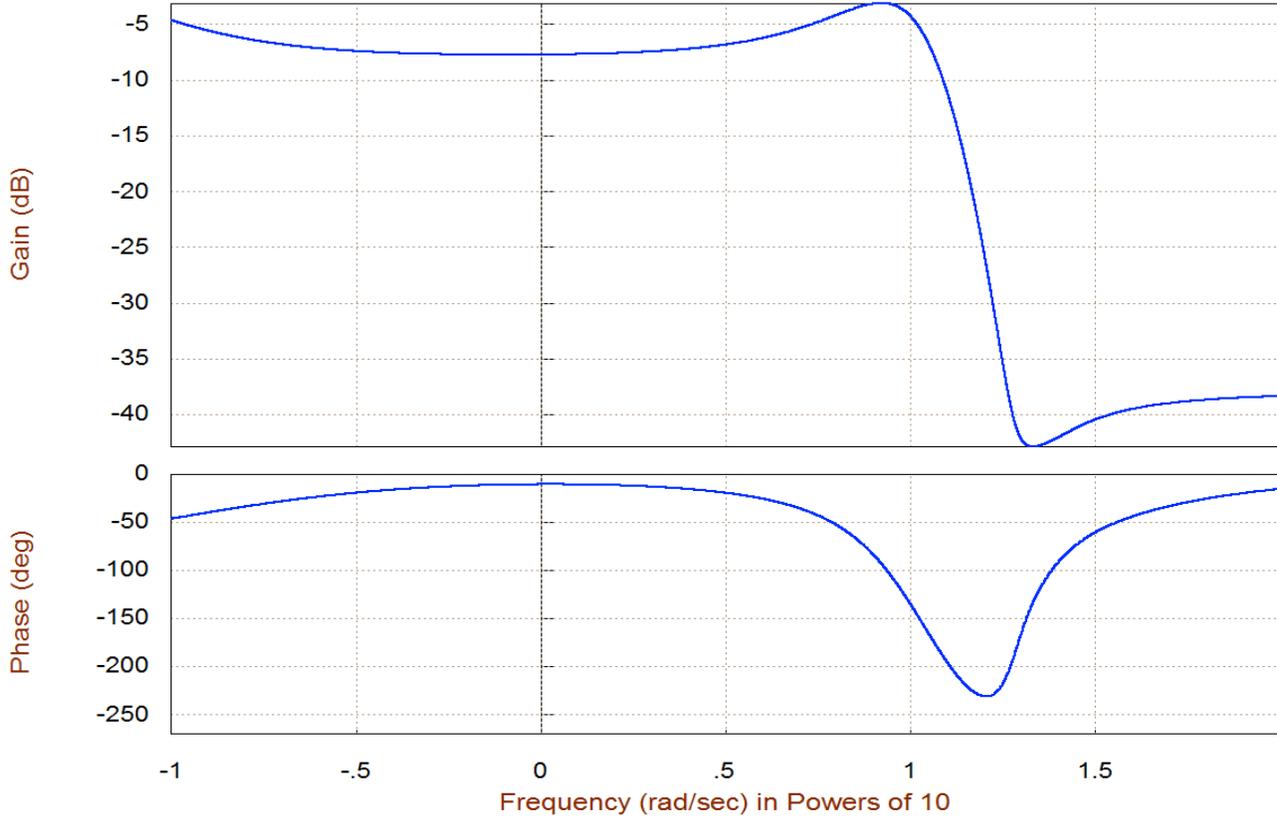
Bode Plot for: Outp(1)-Roll FCS (DP-TVC) / Inpnt(1)-Roll Attitude Error (phi-err) , of: Shuttle Stage-2 Continuous Flight Control System



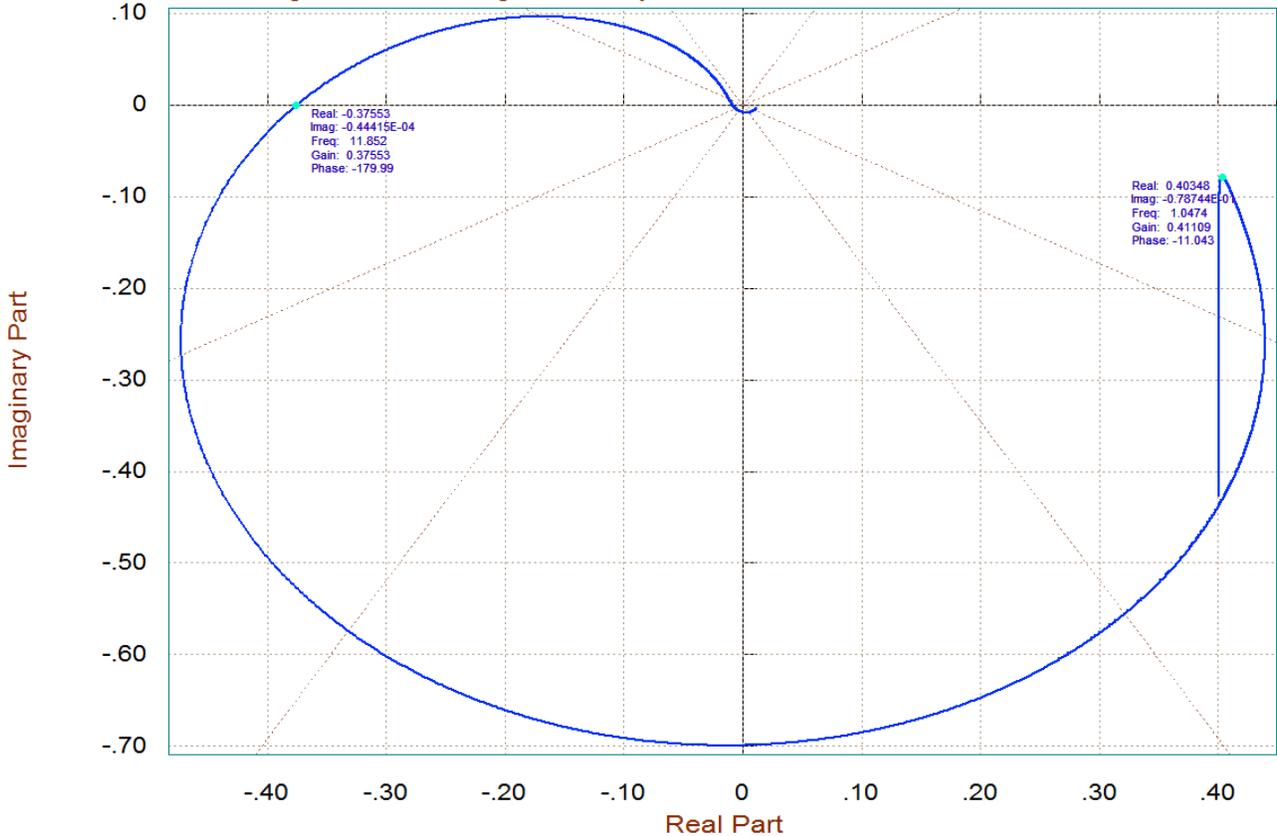
Nyquist Plot for: Outp(1)-Roll FCS (DP-TVC) / Inpnt(1)-Roll Attitude Error (phi-err) , of: Shuttle Stage-2 Continuous Flight Control System



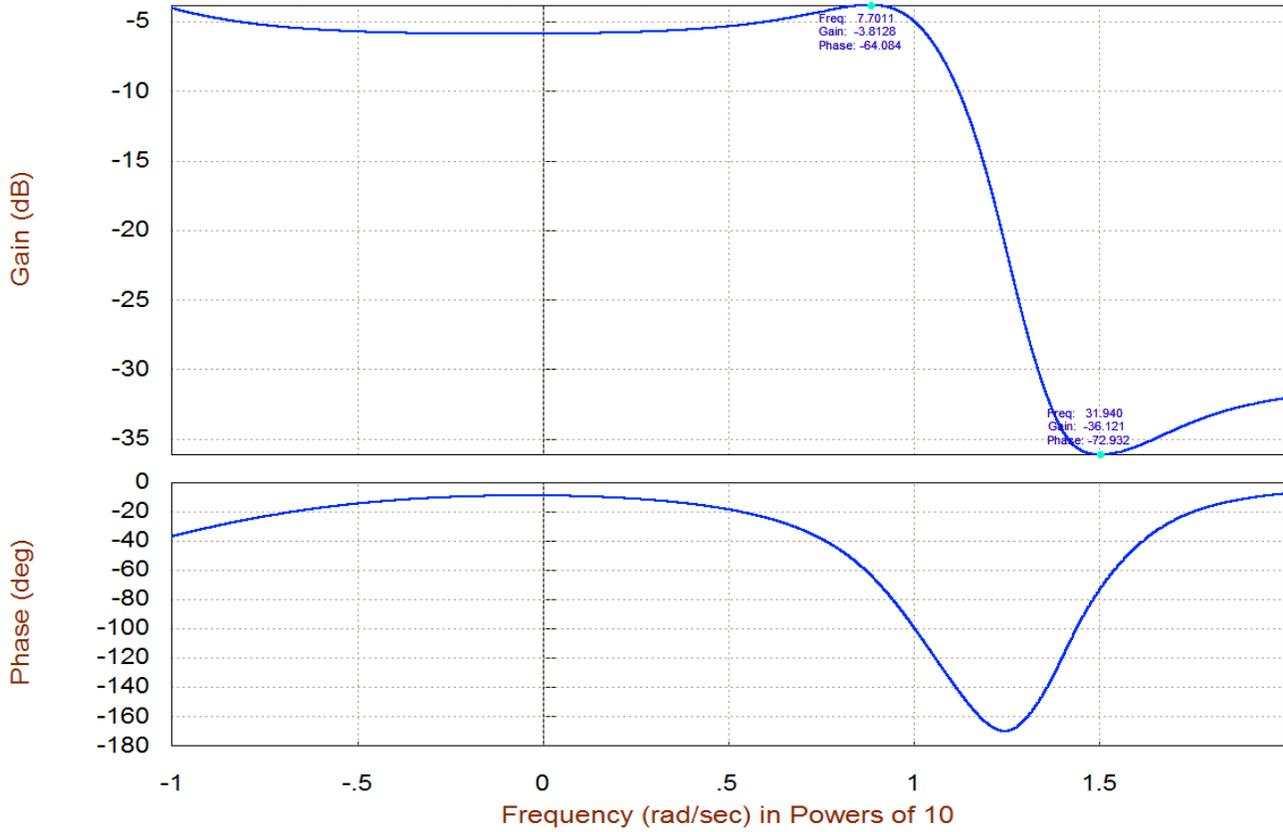
Bode Plot for: Outp(2)-Pitch FCS (DQ-TVC) / Inpt(2)-Pitch Attitude Error (theta-er , of Shuttle Stage-2 Continuous Flight Control System



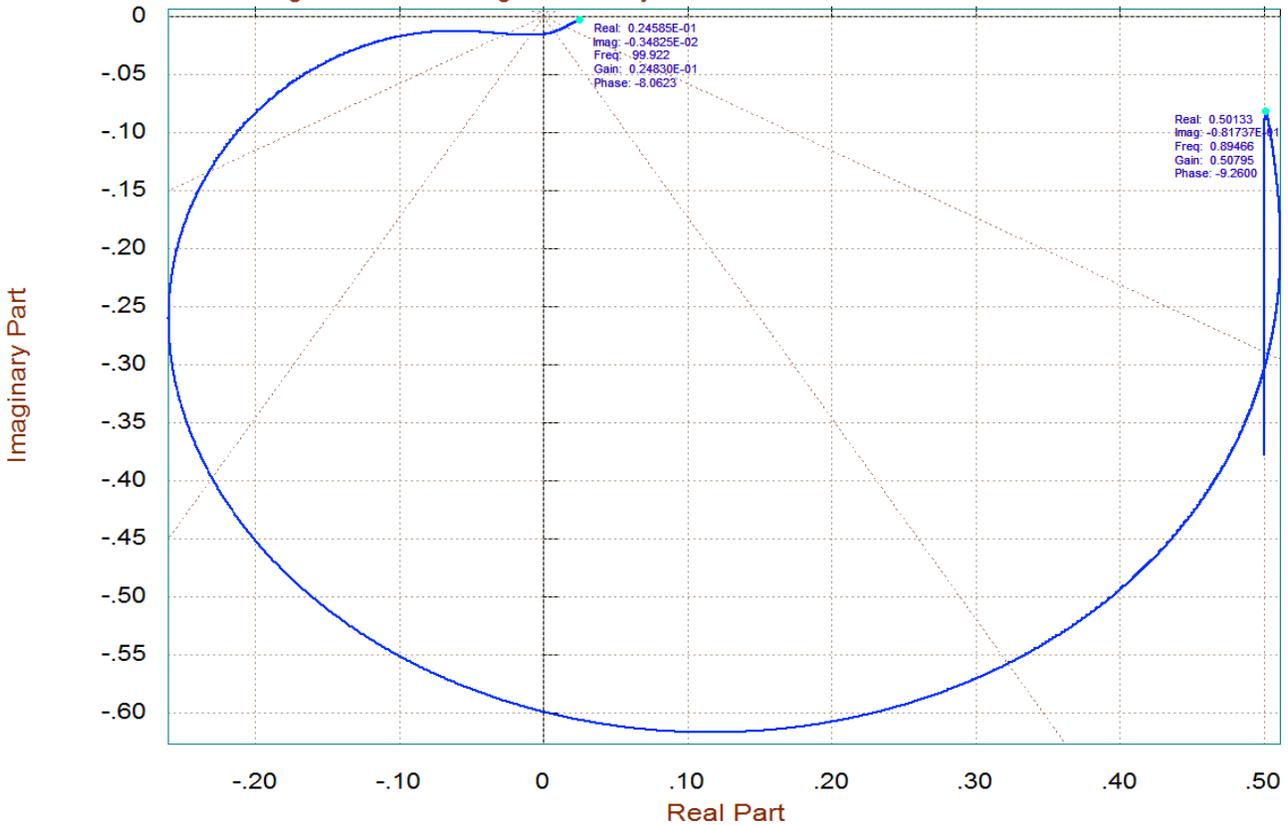
Nyquist Plot for: Outp(2)-Pitch FCS (DQ-TVC) / Inpt(2)-Pitch Attitude Error (theta-er , of Shuttle Stage-2 Continuous Flight Control System



Bode Plot for: Outp(3)-Yaw FCS (DR-TVC) / Inpt(3)-Yaw Attitude Error (psi-err) , of: Shuttle Stage-2 Continuous Flight Control System



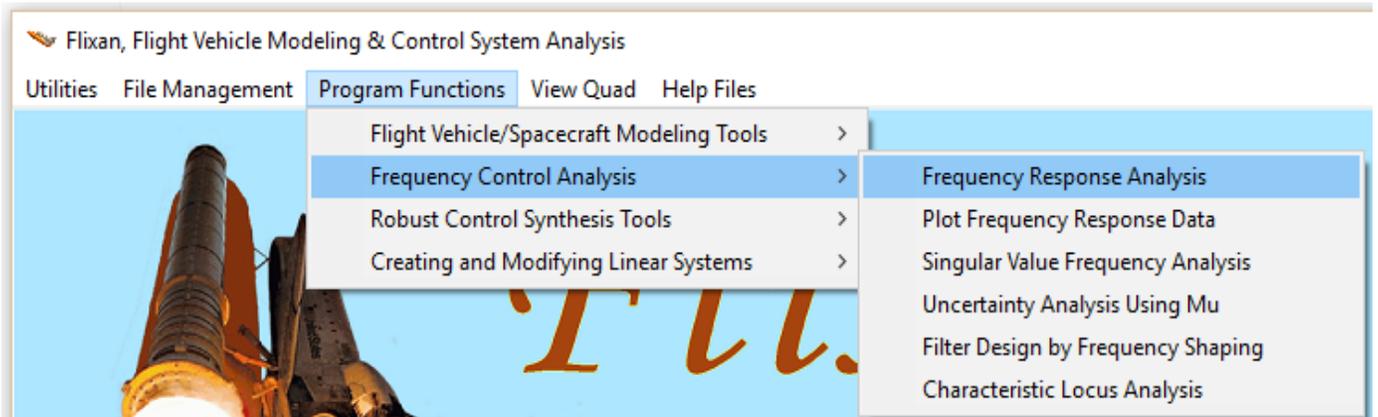
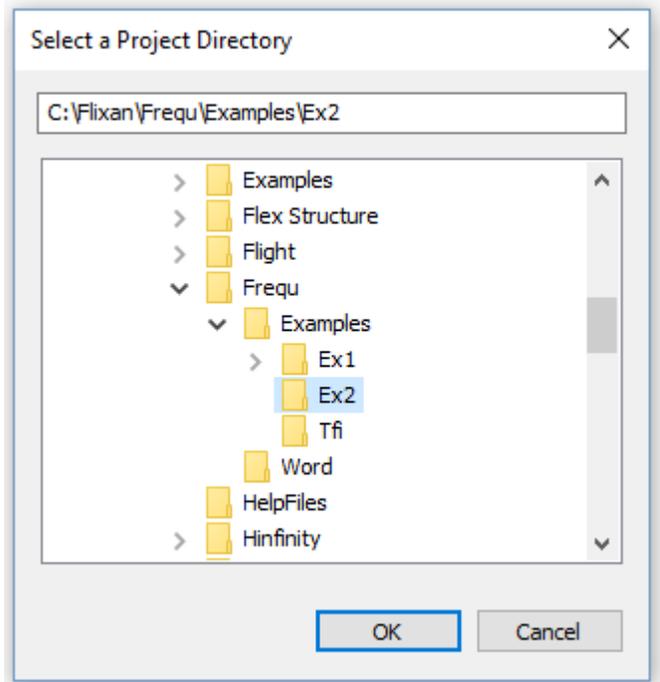
Nyquist Plot for: Outp(3)-Yaw FCS (DR-TVC) / Inpt(3)-Yaw Attitude Error (psi-err) , of: Shuttle Stage-2 Continuous Flight Control System

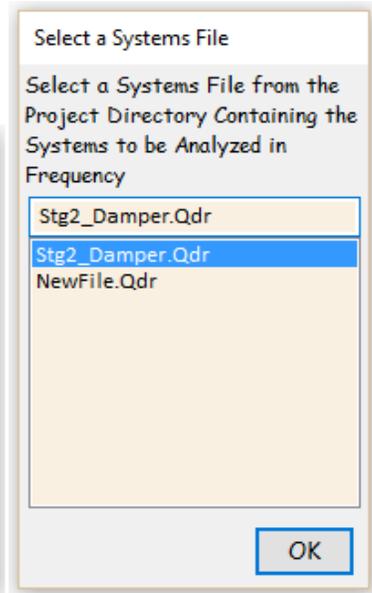
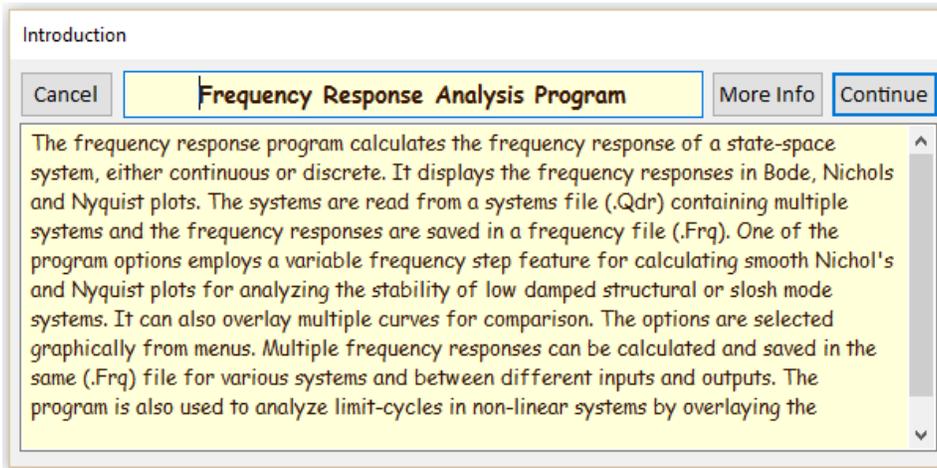


Example 2 Variable Frequency Step/ Overlay

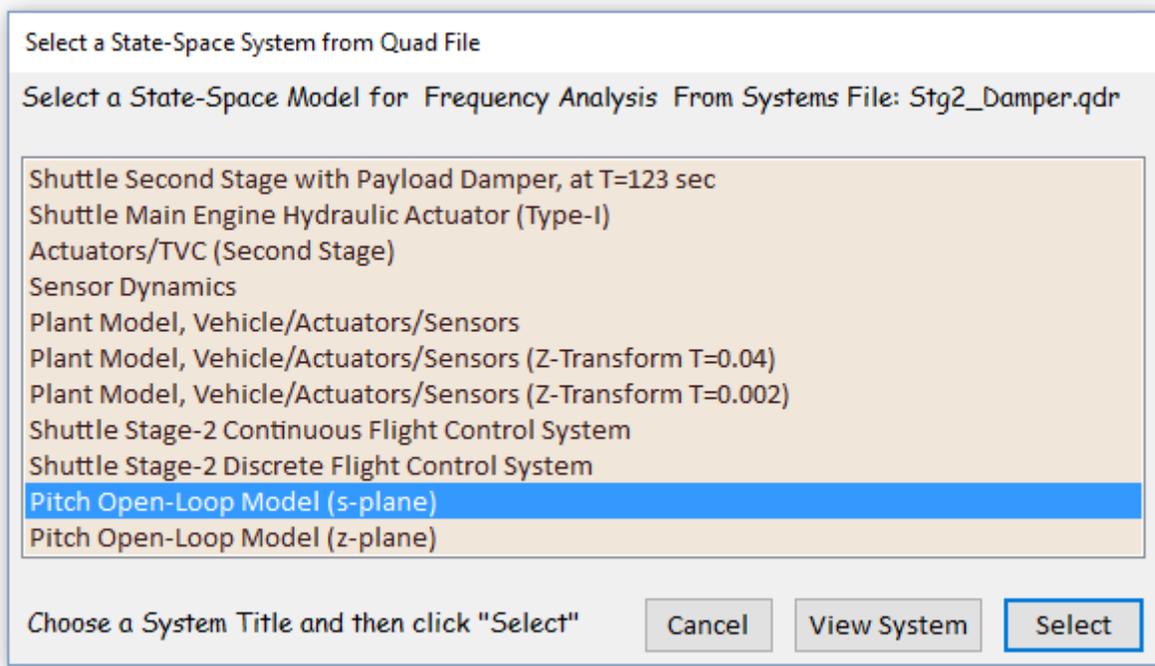
Our second example is in folder “C:\Flixan\Frequ\Examples\Ex2”. In this example we will calculate the frequency response of two similar systems and place them in two separate (.Frq) files in order to overlay them in the same plots for comparison. The two systems represent the open-loop dynamics of the Shuttle vehicle with structural flexibility and propellant sloshing. The actuator, sensor, and control system dynamics are included in the models. The loops are opened in the pitch axis and they will be used for pitch stability analysis. The only difference between the two systems is that the first one is a continuous system that includes a continuous controller, but the second system is discrete that includes a discrete controller. The sampling period of the second system is 40 msec. We will calculate the frequency response of the continuous system and save the response in file vfs1.frq. Then we will calculate the response of the discrete system and save the response in file vfs2.frq. Then we will plot the responses from both systems together to show that the discrete system has some additional attenuation and more phase-lag at high frequencies.

Start the Flixan program, select the project directory and from the main menu select “Program Functions”, “Frequency Control Analysis”, and then “Frequency Response Analysis”. The following is an introduction dialog and click “Continue”. Select also the systems file “Stg2_Damper.Qdr” from the next menu and click “OK”.





From the systems selection menu below, select the continuous system title “*Pitch Open-Loop Model (s-plane)*” and click on “*Select*”.



Use the dialog below to set the initialization parameters for the frequency response calculations of the continuous system, as shown, and click “OK”. Notice that the variable frequency step (VFS) option is selected and the number of points are 10,000.

Frequency Response Program

Initialization Parameters for Frequency Response Cancel OK

System Title:

Enter the Number of Frequency Points

There are two methods of Calculating the Frequency Response, Select Either 1 or 2

Enter an Integer N: (7 to 10) that determines the distance " $10^{(-N)}$ " of Pole/ Zero Cancellation

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems.

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary

Enter Minimum Frequency in (rad/sec)

Enter Maximum Frequency in (rad/sec)

Resolution Parameter for Variable Frequency Step

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

Stability Criteria

Size of the M-circle in Nyquist Diagrams

Gain Margin (dB) in Nichols Charts

Phase Margin (deg) in Nichols Charts

Stg2_Damper-1.Frq

Stg2_Damper.Frq

From the following input/ output selection menu, select the single input and the single output to calculate the frequency response of the system that has the loop opened at the input of the pitch actuator, and click "OK".

Select Input/ Output Transfer ×

System Title:

Select an Input and an Output from below, for Calculating the System Frequency Response OK

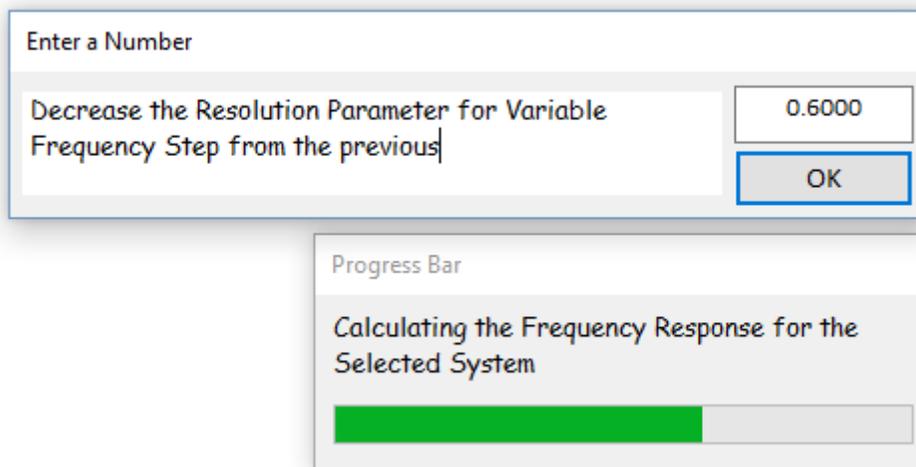
System Inputs:

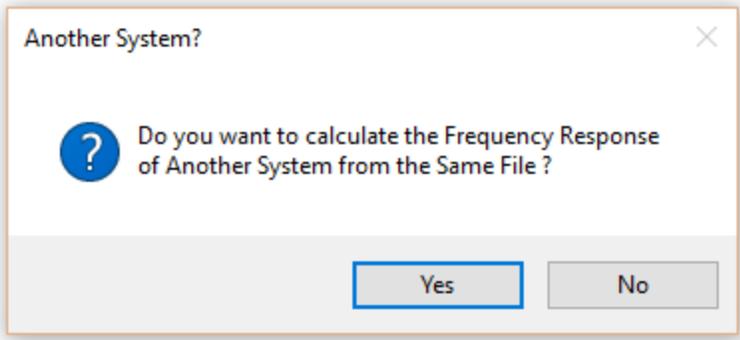
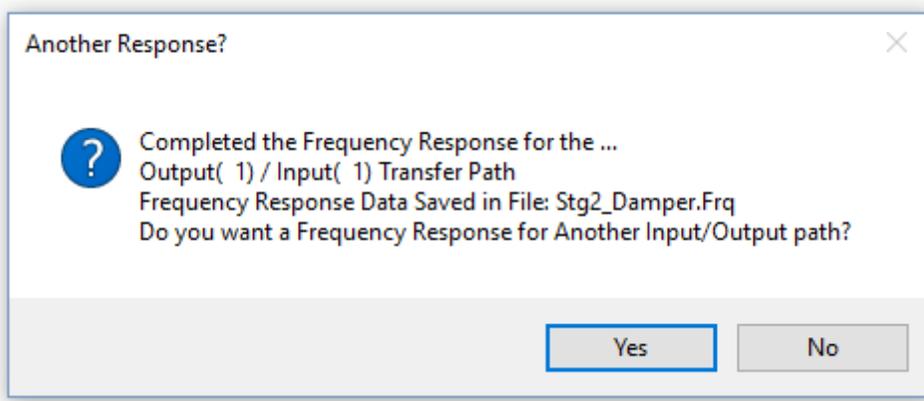
System Outputs:

When VFS is selected the algorithm adjusts the frequency step in order to generate smooth Nichol's and Nyquist plots. This feature is useful when analyzing systems that include low damping resonances such as slosh and structural modes. The smoothness of the Nichol's and Nyquist plots depends on the number of points used. If only a few points are used the modes will appear like polygons instead of circular. The VFS algorithm is attempting to adjust the frequency step between calculations in order to keep a certain variable as close to the resolution parameter as possible. This criterion variable is a combination of three terms $\{\delta(\text{phase}) + \delta(\text{gain}) + K \delta(\text{frequency})\}$. That is, a change in gain, phase, and frequency which is maintained almost constant between subsequent calculations.

It is not easy to set the frequency range exactly as desired when using the VFS option and several attempts are needed in order to achieve the specified final frequency. The resolution parameter controls the spacing between the frequency points and it is typically set between (0.4 and 0.9). If the value of the resolution parameter is small there will be more points used to describe each resonance and the final frequency for a fixed number of points will be small. When a large resolution is used there will be less points used per resonance, and the final frequency will be higher than expected for the same number of points. The frequency range is, therefore, controlled by selecting the proper resolution parameter value depending on the number of modes and the number of points selected. This may require several trials of adjusting the resolution parameter in order to cover the desired frequency range.

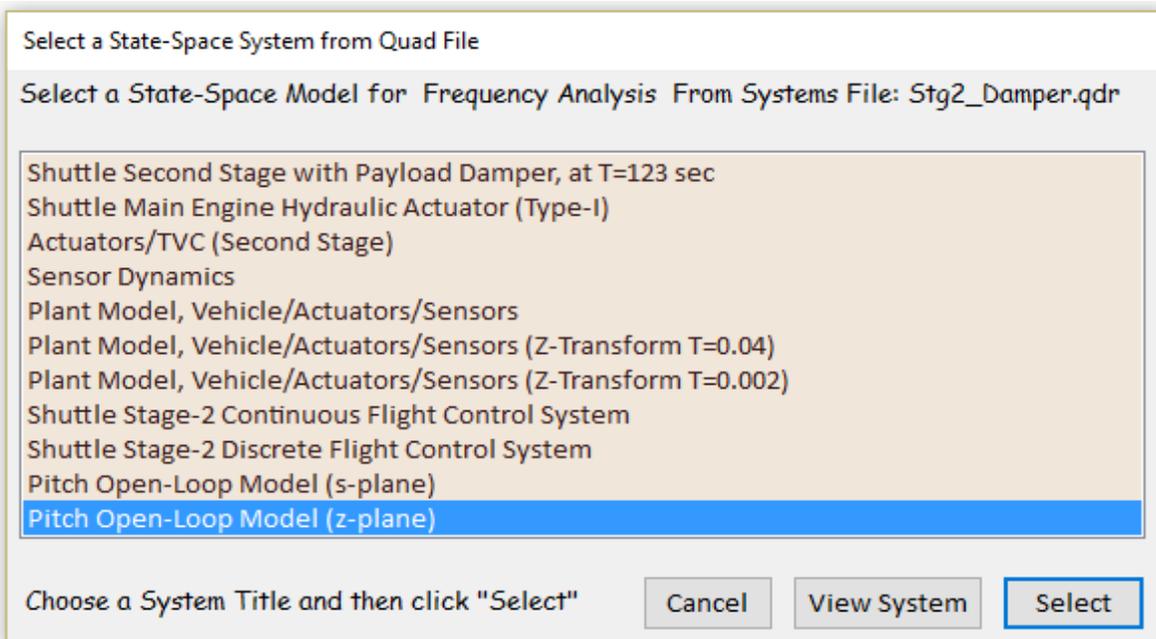
The program will assist the user in the process of selecting the resolution parameter. For example, if you begin with a resolution value of 0.6, and if the final frequency is a lot smaller than the specified range, the program will ask you to increase the resolution parameter. If the final frequency exceeds the specified final frequency by a significant amount, the program will ask you to reduce the resolution and repeat the process. The frequency at the final point will be acceptable when it falls close to 80% of the specified final frequency, otherwise, you will be prompted to adjust the resolution parameter. In this case the resolution parameter was reduced to 0.35 in order to get near the specified final frequency which is 100 (rad/sec).





Click on “No” in both of the above dialogs and the program will plot the frequency response data which are saved in file “Stg2_Damper.Frq”. This file is also saved under a different name “vfs1.frq” to avoid being overwritten.

Do not plot the data yet but repeat the frequency response calculation process as before, by selecting the same systems filename “Stg2_Damper.Qdr” and then from the systems selection menu, select the discrete system title “Pitch Open-Loop Model (z-plane)” and click on “Select”.



Initialize the frequency response calculations parameters for of the discrete system, as shown below, and click “OK”. Notice that this time we are going to use the same frequency points that were used in the previously calculated file “vfs1.frq”. Then from the following menu select the single input and the single output of that system.

Frequency Response Program

Initialization Parameters for Frequency Response Cancel OK

System Title: **Pitch Open-Loop Model (z-plane)**

Enter the Number of Frequency Points: 10000

There are two methods of Calculating the Frequency Response, Select Either 1 or 2: 1

Enter an Integer N: (7 to 10) that determines the distance "10^{-N}" of Pole/ Zero Cancellation: 9

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method: **SISO**

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems: **Yes**

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary: **Variable**

Enter Minimum Frequency in (rad/sec): 0.10000

Enter Maximum Frequency in (rad/sec): 100.00

Resolution Parameter for Variable Frequency Step: 0.60000

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

Stability Criteria

Size of the M-circle in Nyquist Diagrams: 2.0000

Gain Margin (dB) in Nichols Charts: 8.0000

Phase Margin (deg) in Nichols Charts: 40.000

vfs1.Frq

Stg2_Damper-1.Frq

Stg2_Damper.Frq

vfs1.Frq

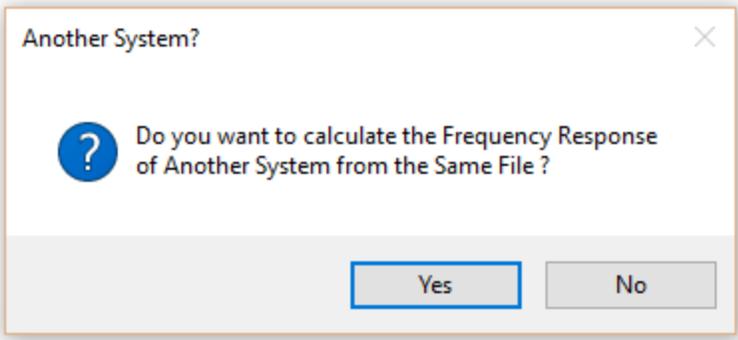
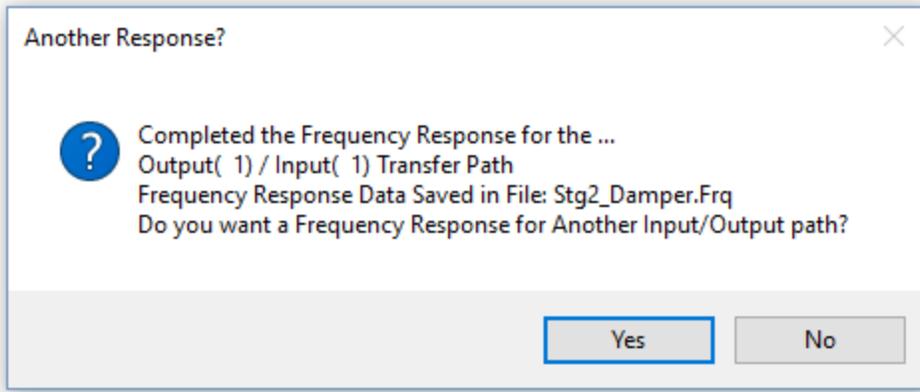
Select Input/ Output Transfer ×

System Title: **Pitch Open-Loop Model (z-plane)**

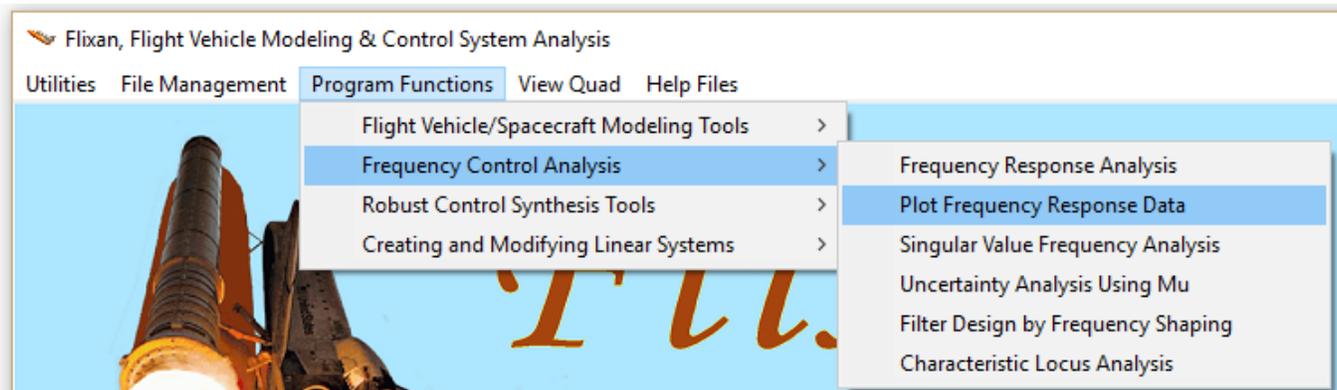
Select an Input and an Output from below, for Calculating the System Frequency Response OK

System Inputs: System Outputs:

Pitch FCS Command (DQ-TVC) Pitch FCS Command (DQ-TVC)



Click on “No” in both of the above dialogs and the frequency response data will be saved in file “Stg2_Damper.Frq”. This file is also saved under a different name “vfs2.frq” to avoid being overwritten. You may now plot the frequency response data from the two files. Go to the Flixan main menu and select “Program Functions”, “Frequency Control Analysis”, and then “Plot Frequency Response Data”.



From the top two menus of the following dialog select the two files “vfs1.frq” and “vfs2.frq” containing the frequency responses of the two systems and click “OK”. The third menu is used for selecting a Describing Function file which is applicable only for non-linear system analysis. Use the main menu of the post-processing program to generate Bode, Nichols, and Nyquist plots as shown.

Select two Frequency Files

Enter One or Two previously calculated Frequency Response files (*.Frq) to Plot and Overlay for Comparison

vfs1.Frq	vfs2.Frq
Stg2_Damper-1.Frq	Stg2_Damper-1.Frq
vfs1.Frq	vfs1.Frq
vfs2.Frq	vfs2.Frq

You May also select a Describing Function File (*.DF) to Overlay for Non-Linear Limit-Cycle Analysis ---->

OK

Main Frequency Response Menu

Select one of the following options

- Gain and Phase versus Frequency, Bode Plots
- Gain versus Phase, Nichols Plots
- Real versus Imaginary Parts, Nyquist Plots
- Read the Next Set of Frequency Data
- Go Back to the Beginning of the Frequency File
- Exit the Frequency Plotting Program

Main Frequency Response Menu

Select one of the following options

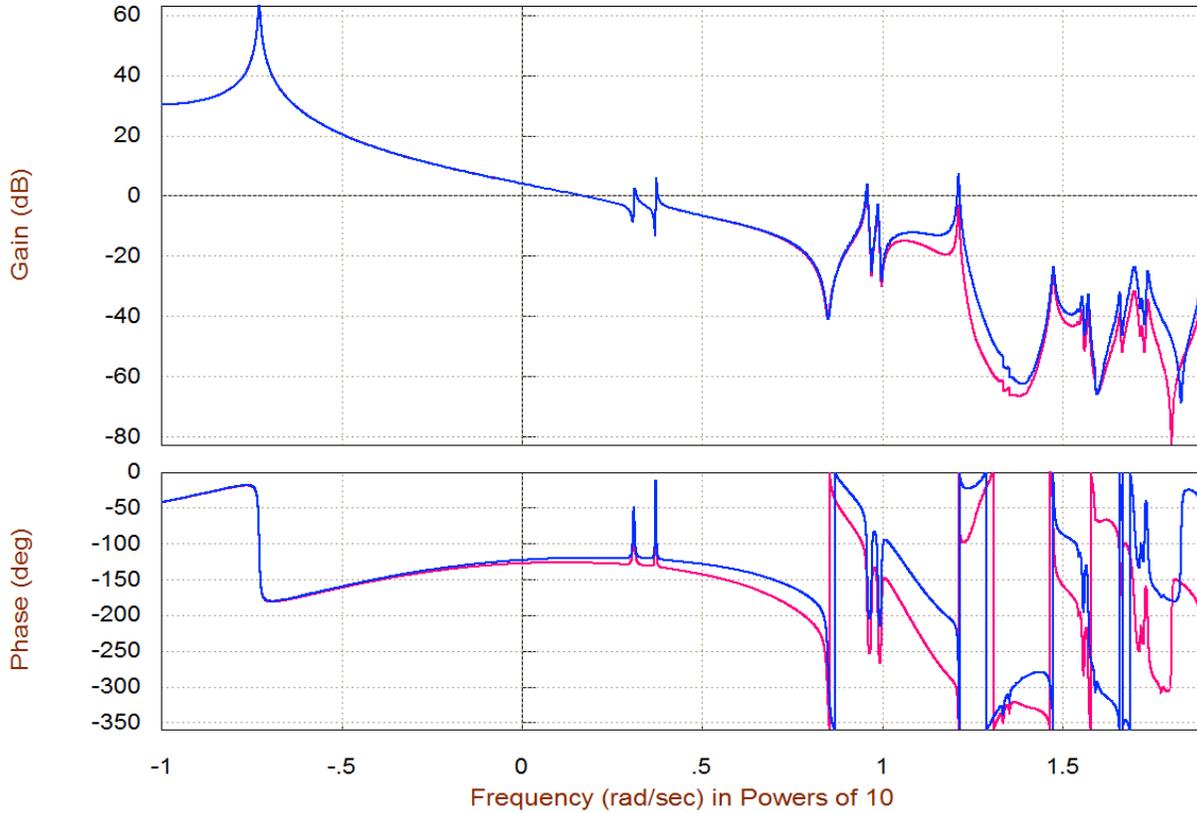
- Gain and Phase versus Frequency, Bode Plots
- Gain versus Phase, Nichols Plots
- Real versus Imaginary Parts, Nyquist Plots
- Read the Next Set of Frequency Data
- Go Back to the Beginning of the Frequency File
- Exit the Frequency Plotting Program

Main Frequency Response Menu

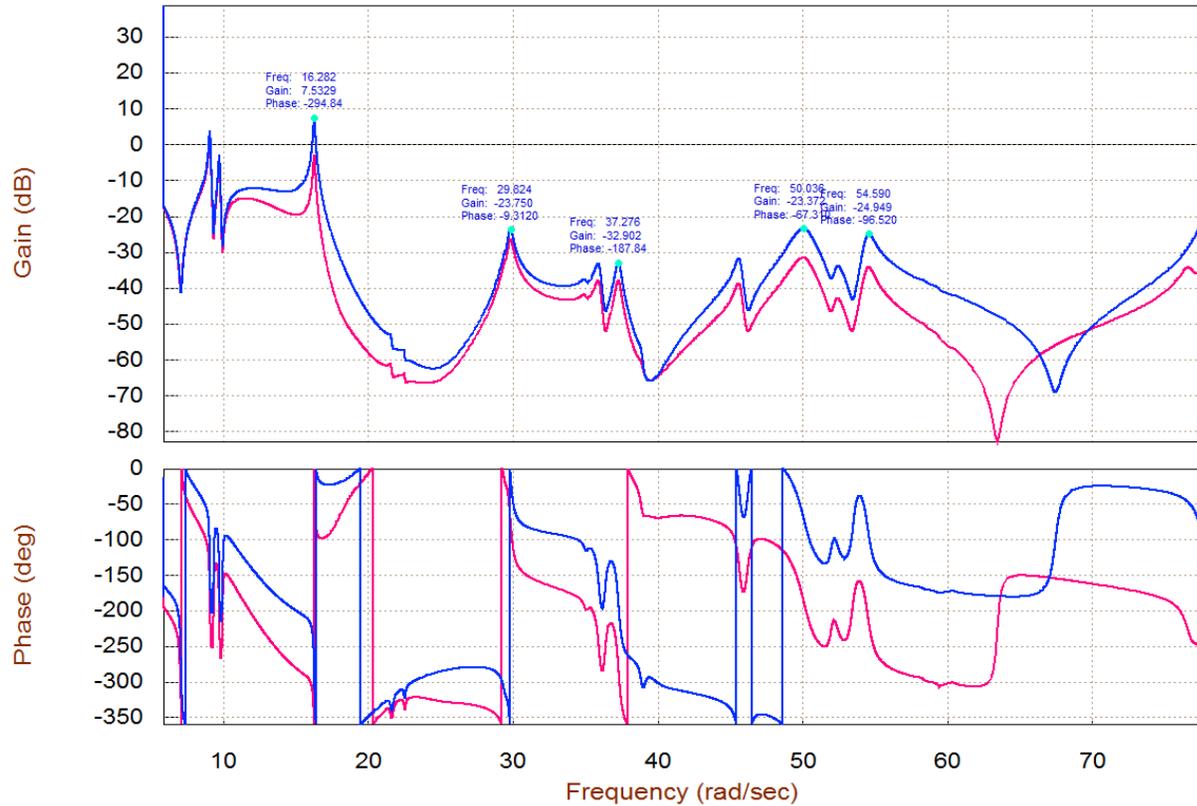
Select one of the following options

- Gain and Phase versus Frequency, Bode Plots
- Gain versus Phase, Nichols Plots
- Real versus Imaginary Parts, Nyquist Plots
- Read the Next Set of Frequency Data
- Go Back to the Beginning of the Frequency File
- Exit the Frequency Plotting Program

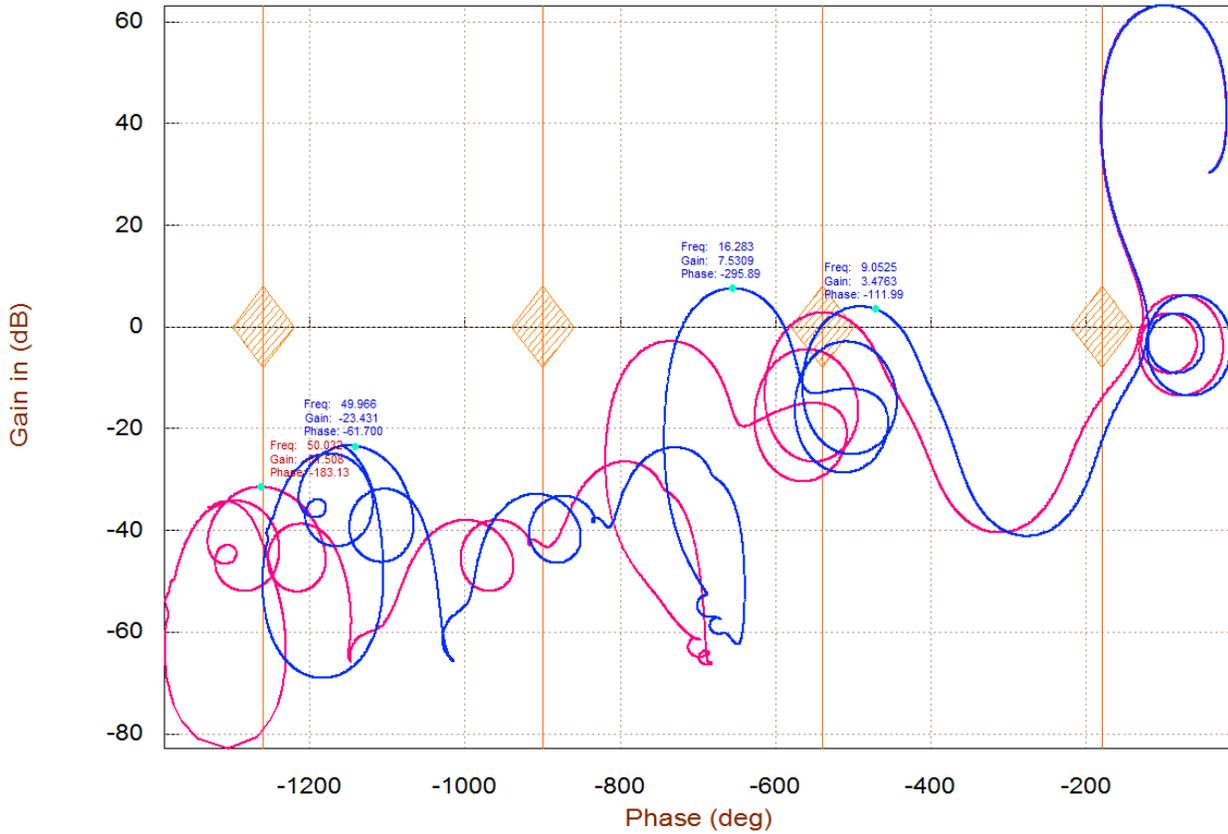
Bode Plot for: Outp(1)-Pitch FCS Command (DQ-TVC) / Inpt(1)-Pitch FCS Command (DQ-TVC) , of: Pitch Open-Loop Model (z-plane)



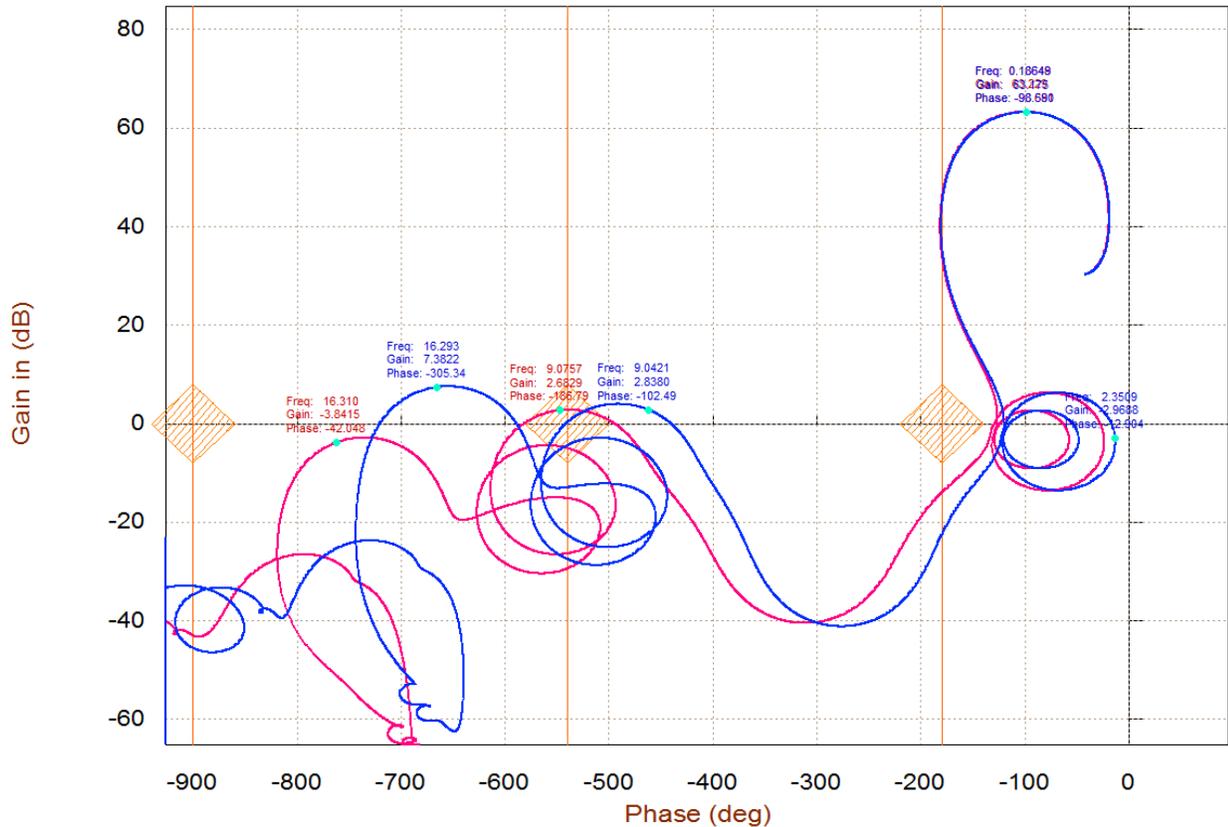
Bode Plot for: Outp(1)-Pitch FCS Command (DQ-TVC) / Inpt(1)-Pitch FCS Command (DQ-TVC) , of: Pitch Open-Loop Model (z-plane)



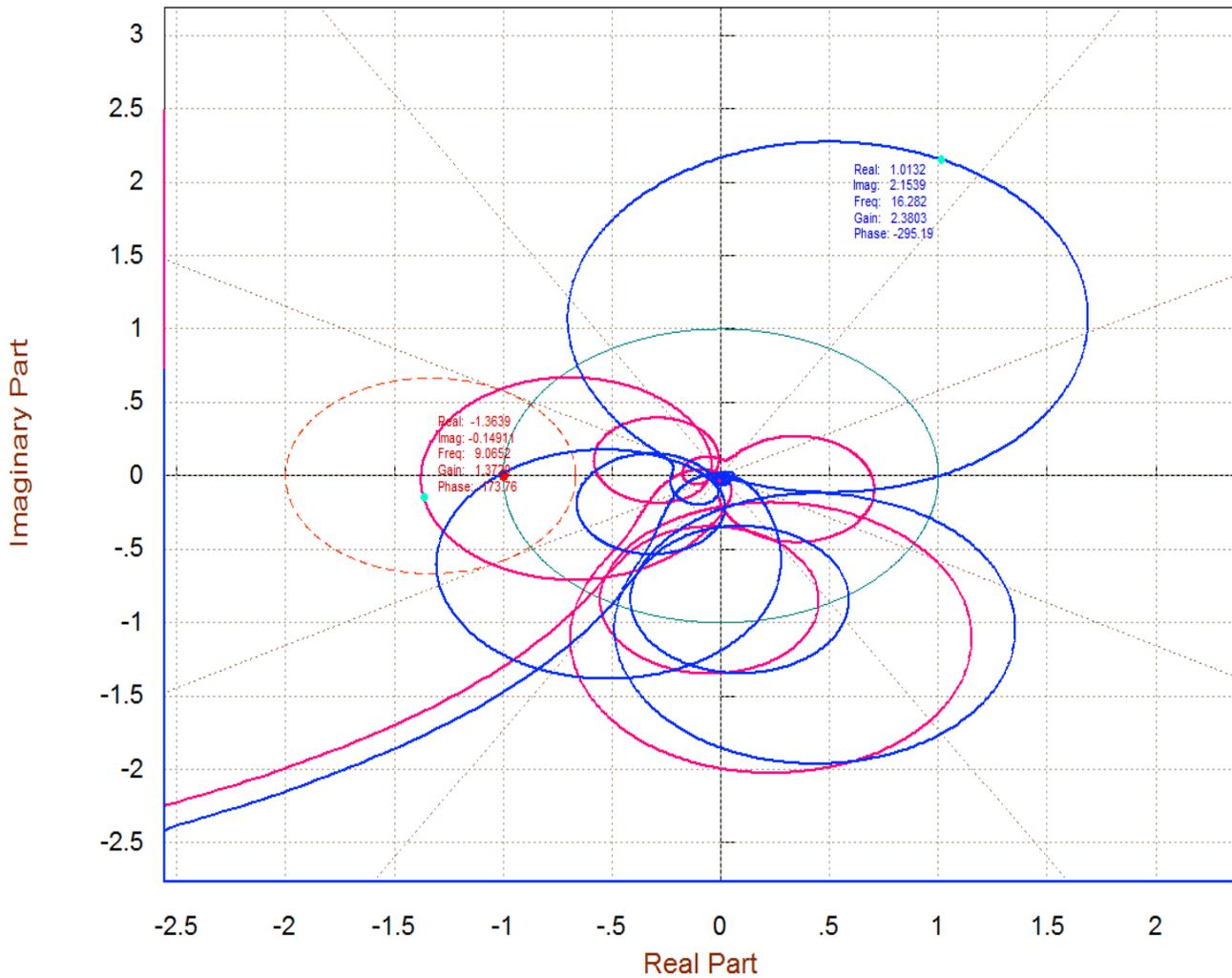
Nichols Plot for: Outp(1)-Pitch FCS Command (DQ-TVC) / Inpt(1)-Pitch FCS Command (DQ-TVC) , of: Pitch Open-Loop Model (z-plane)



Nichols Plot for: Outp(1)-Pitch FCS Command (DQ-TVC) / Inpt(1)-Pitch FCS Command (DQ-TVC) , of: Pitch Open-Loop Model (z-plane)



Nyquist Plot for: Outp(1)-Pitch FCS Command (DQ-TVC) / Inpt(1)-Pitch FCS Command (DQ-TVC) , of: Pitch Open-Loop Model (z-plane)



Notice how the discrete system's response (magenta colour), although similar in nature, it has some additional phase-lag and attenuation which is more observable at high frequencies. It is caused mainly by the zero-order-hold. Notice also that one of the structural modes, at 9 (rad/sec) is unstable because it encircles the critical point. This is noticeable in both Nichols and Nyquist plots. If not attenuated, this structural mode will create divergent structural oscillations at 9 (rad/sec). This can be observed in closed-loop simulations.

Example 3 Describing Function Analysis

In this example we will use the Describing Function (DF) method to analyze the structural stability of the Space Shuttle payload. The Shuttle is carrying a flexible payload which is rigidly attached to the cargo bay at aft end. The front section of the payload, however, is attached to the cargo bay by means of two Coulomb Dampers which are non-linear devices, as shown in Figure 1. The purpose of the Coulomb Dampers is to attenuate oscillations and to reduce disturbances on the payload by dissipating structural energy.

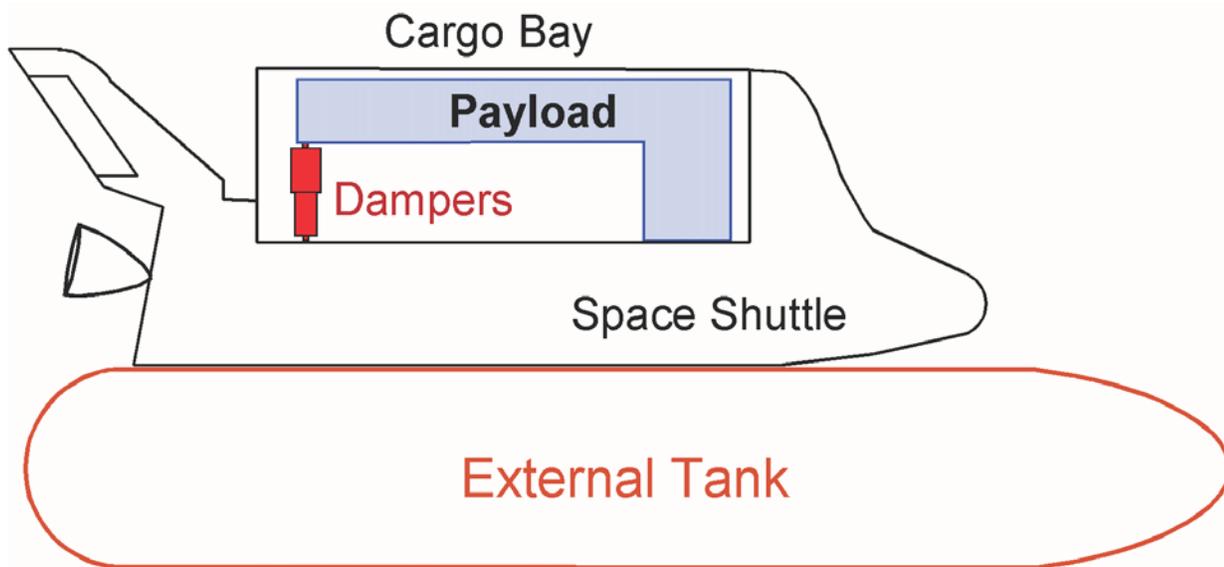


Figure 3.1 Payload is rigidly attached to the Shuttle on one side and on the other side it is attached to the Cargo Bay by two Coulomb Dampers

In this example we will analyze the dynamic interaction between the flexible structure and the non-linear Coulomb dampers by frequency domain analysis and simulation. We will use the Describing Function (DF) method to estimate the size and frequency of the limit-cycles caused by the non-linear devices, and simulations to validate the stability analysis. The files for this Shuttle second stage analysis example are located in directory "C:\Flixan\Frequ\Examples\Ex3". The input data file is "Damper_DF.Inp" and the systems file is "Damper_DF.Qdr" containing the vehicle state-space systems. The DF of the Coulomb damper is already calculated and saved in file "Coulomb.DF".

The systems file includes the system "Pitch DF Analysis Model (s-plane)". This system consists of the flexible vehicle, sensor and actuator dynamics and the flight control system with the 3 loops closed, as shown in the block diagram Figure 3.2. The mechanical loop, however, across the damper is opened for frequency response analysis. This system is used for analyzing stability in the pitch direction by exciting and measuring the structural modes symmetrically. The input is force applied across the left and right damper in (lb), and the output is the average displacement across the two dampers. A similar model is used for analyzing stability in the anti-symmetric direction by applying equal and opposite forces on the left and right dampers and measuring the differential displacements of the two dampers. This example is presented in more detail in "Flixan\Examples\Payload Damper".

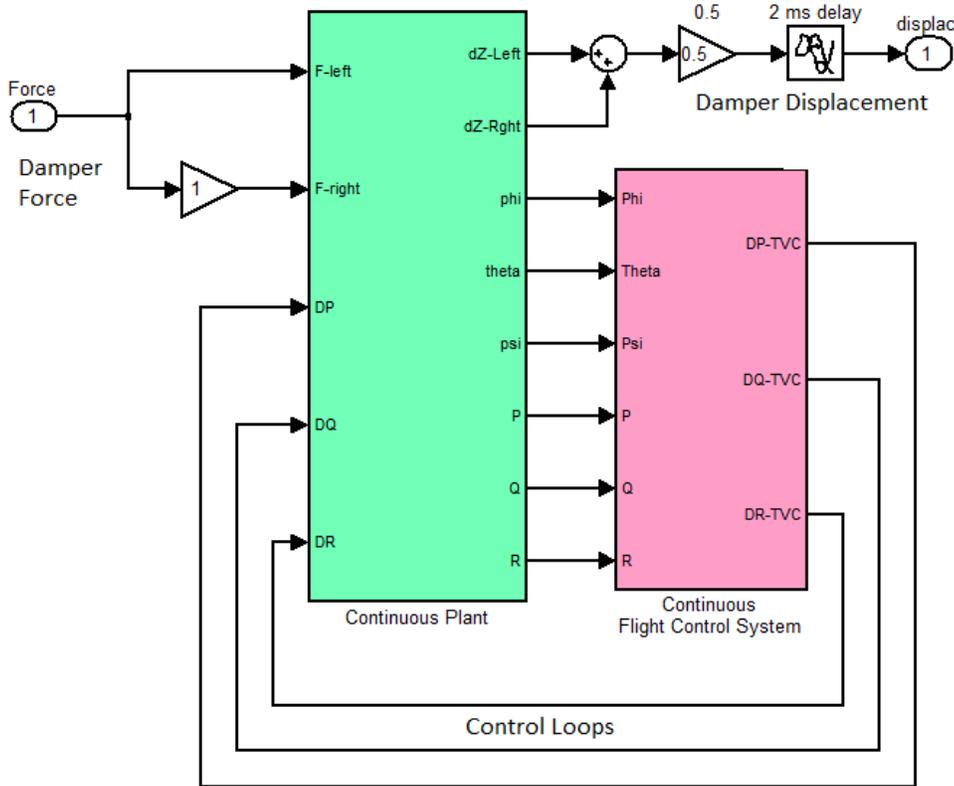
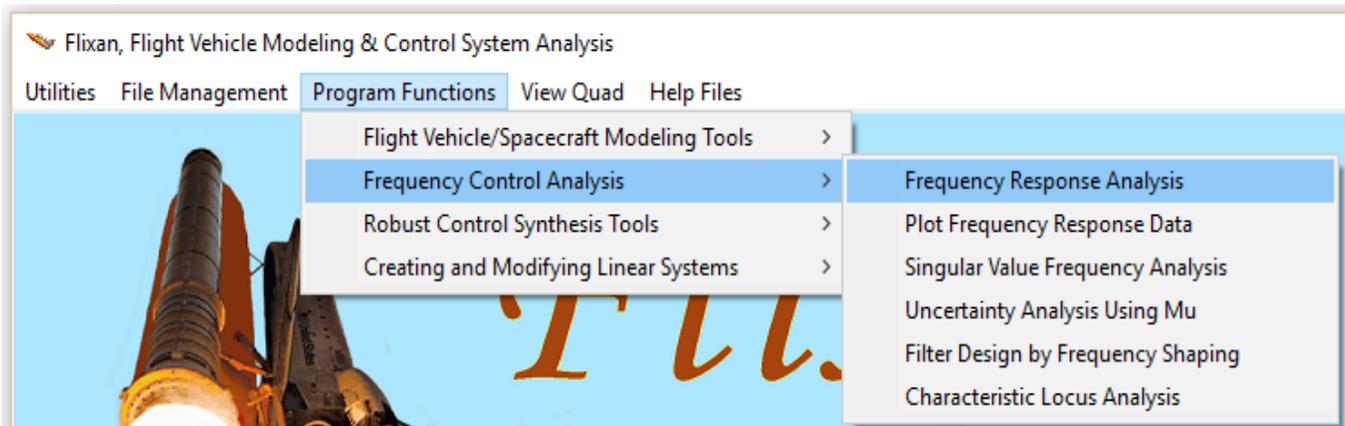
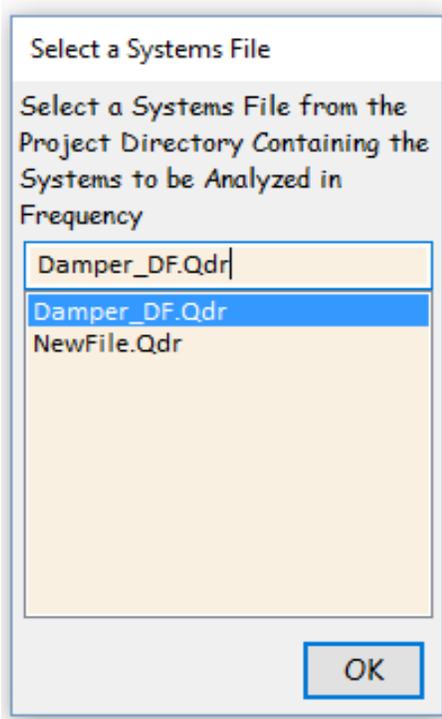


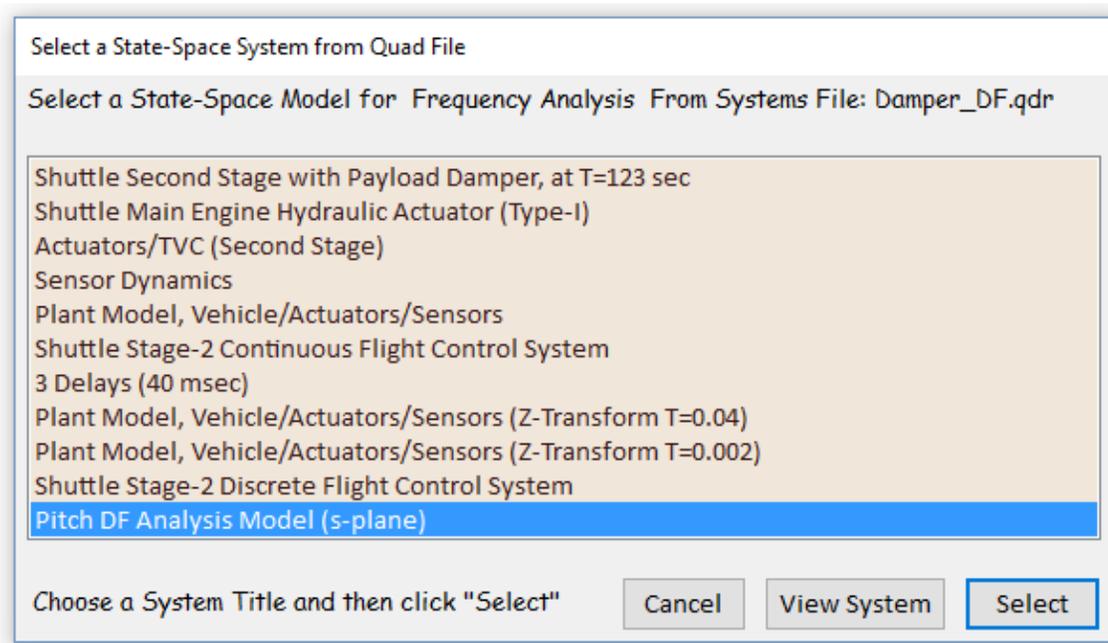
Figure 3.2 Pitch DF Analysis Model; The Flight Control Loop is Closed and the Loop is Opened across the Dampers

Start the Flixan program, select the project directory and from the main menu select “Program Functions”, “Frequency Control Analysis”, and then “Frequency Response Analysis”. From the systems filename selection menu select the systems file “Damper_DF.Qdr”, and click “OK”.





From the following menu select the system *"Pitch DF Analysis Model (s-plane)"*. This system has the flight control loops closed, as already described in Figure 3.2, and the mechanical loop across the damper is opened for frequency response analysis. Click on "Select" and the parameters initialization dialog opens up, as shown below. Use the variable frequency step with 10,000 points to obtain smooth resonances. Then select the only one input and the only one output corresponding to the force on the vehicle generated by the damper and the displacement across the damper.



Frequency Response Program

Initialization Parameters for Frequency Response

Cancel

OK

System Title:

Pitch DF Analysis Model (s-plane)

Enter the Number of Frequency Points

10000

There are two methods of Calculating the Frequency Response, Select Either 1 or 2

1

Enter an Integer N: (7 to 10) that determines the distance "10^(-N)" of Pole/ Zero Cancellation

9

This Option Calculates the Frequency response from Multiple Inputs to Multiple Outputs. It is used by the INA Method

SISO

MIMO

Do you want to Include the ZOH dynamics in the Output? Used only in Discrete Systems.

Yes

No

"Fixed" or "Variable" Frequency Step. Variable Step creates High Resolution Nichols/ Nyquist Plots. Max Frequency may Vary

Fixed

Variable

Enter Minimum Frequency in (rad/sec)

0.10000E-01

Enter Maximum Frequency in (rad/sec)

100.00

Resolution Parameter for Variable Frequency Step

0.22000

Otherwise, You may choose the same frequency points from a Frequency Response File (*.Frq) selected from the Menu on the Right -->

Stability Criteria

Size of the M-circle in Nyquist Diagrams

2.0000

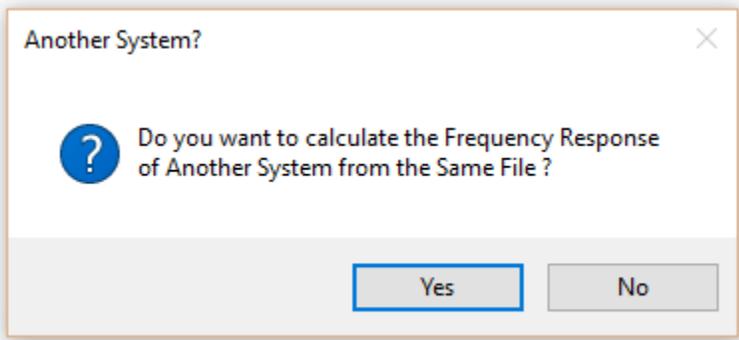
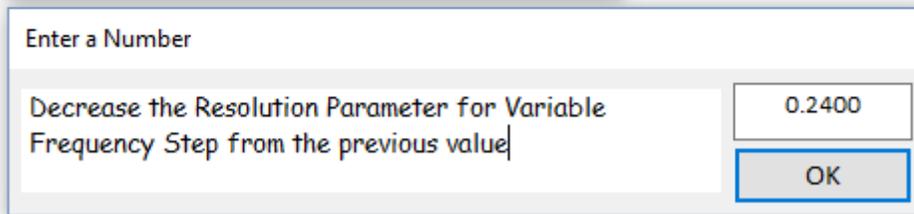
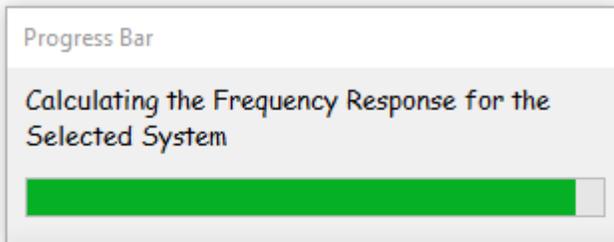
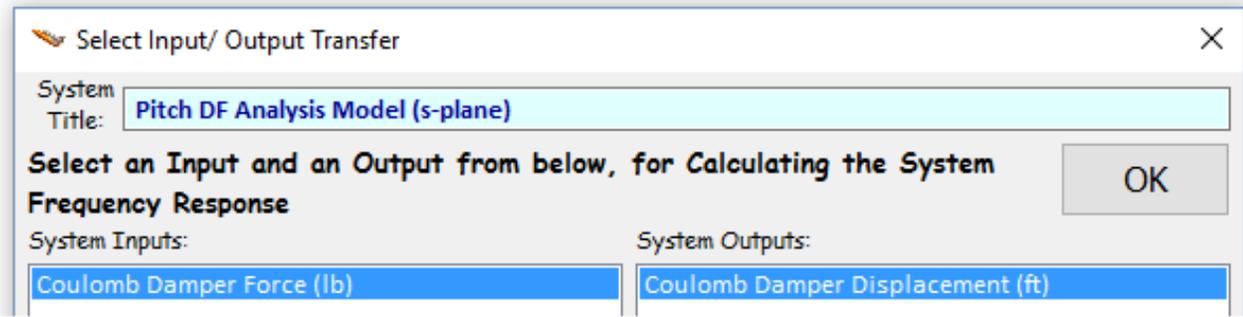
Gain Margin (dB) in Nichols Charts

8.0000

Phase Margin (deg) in Nichols Charts

40.000

Damper_DF-1.Frq
Damper_DF-2.Frq
Damper_DF-3.Frq
Damper_DF-4.Frq
Damper_DF-5.Frq
Damper_DF.Frq



Do not calculate any more frequency responses from other systems. From the following filename selection menus do not select another frequency response file, but select the non-linearity file "Coulomb.DF" that includes the DF data for the Coulomb damper and click on "OK". This file includes sinusoidal input amplitude in (feet) versus gain, and phase in (deg).

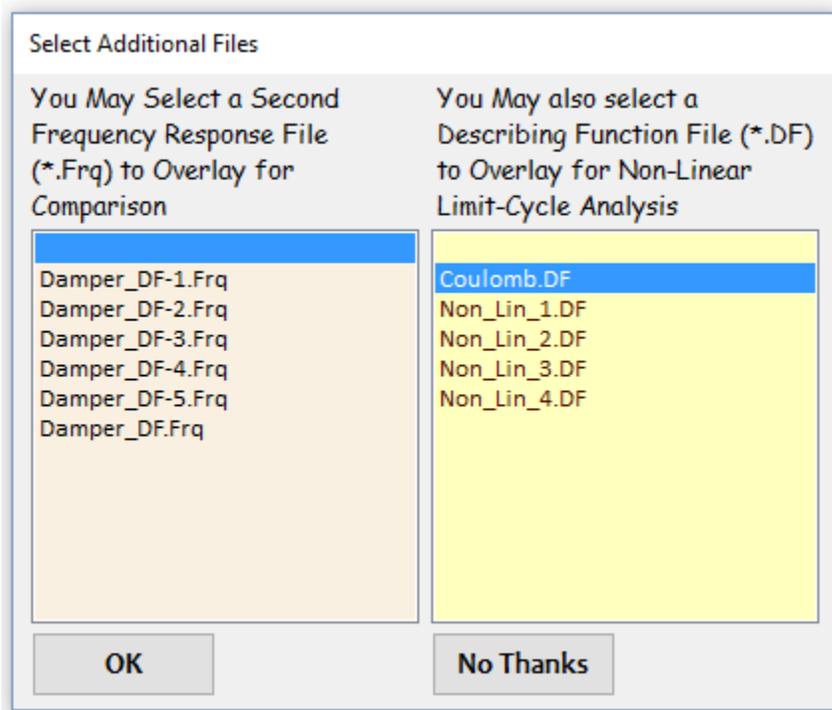
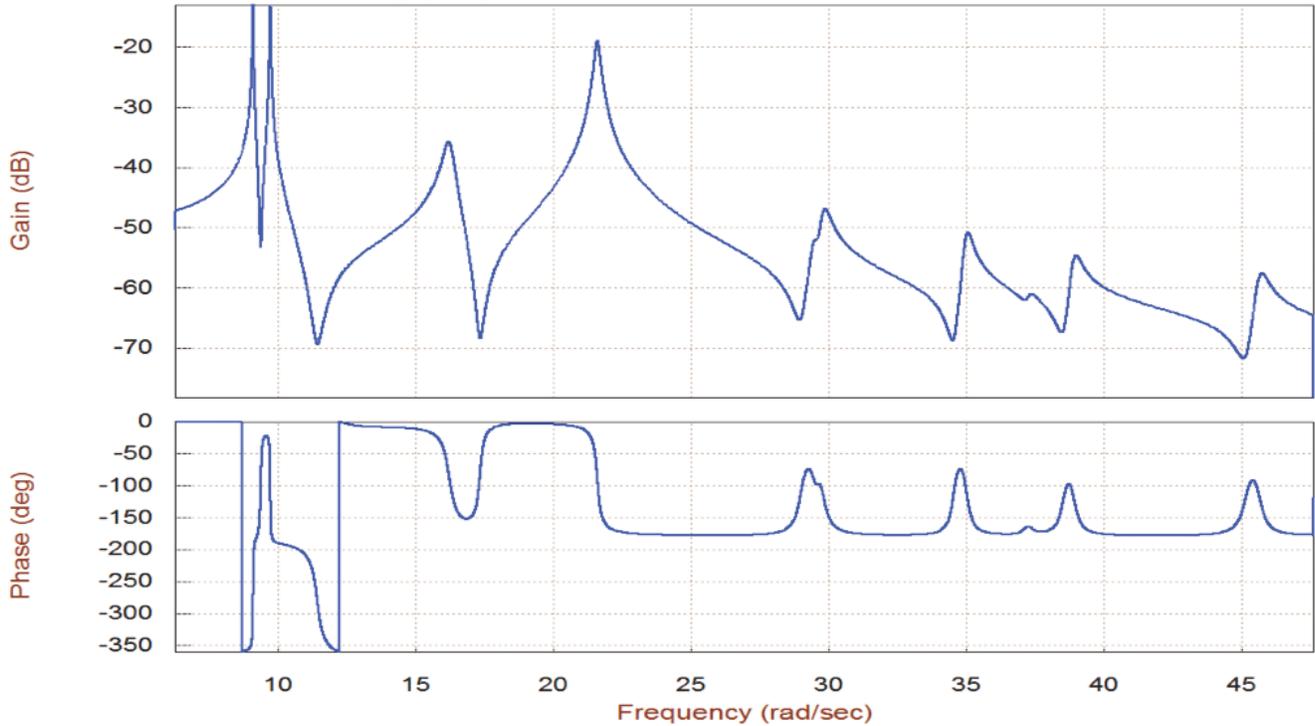


Figure 3.3a shows the Bode frequency response of the symmetric closed-loop plant $G_s(s)$ calculated across the Coulomb damper non-linearity. The excitation forces in (lb) are applied on the vehicle structure in phase at the two damper attachments in order to excite the pitch flexibility. The output of $G_s(s)$ is the average displacement at the two dampers. It shows that the structure has two big resonances at around 9 (rad/sec). The Nichols plot in Figure 3.3b shows the overlay of two loci, the symmetric system $G_s(j\omega)$ co-plotted with the inverse of the damper DF which is $-1/N(a)$. The $-1/N(a)$ locus is shown twice because it repeats every 360° similar to the (+) point in the classical Nichols charts. The intersections indicate that according to the Nyquist criteria there is a convergent limit cycle at 10.5 (rad/sec) indicating that the system will maintain a symmetric oscillation at 10.5 (rad/sec) at an amplitude of 0.004 (ft). There is also a convergent limit-cycle at a lower amplitude of 0.0013 (feet) which is at 9.78 (rad/sec) frequency. This intersection point is in the vertical $\pm 180^\circ$ section of the $1/N(a)$ describing function locus which corresponds to the device operating in the spring/ dead-zone region before breakout. The amplitude of the Coulomb damper oscillation (zero to peak) is obtained from the amplitude of the DF locus at the intersection with the plant $G_s(j\omega)$ locus. There is also a divergent (shown in green) intersection point between the two convergent limit-cycles. The loci are almost touching at a higher frequency which indicates the possibility of a limit cycle at 24 (rad/sec). However, it is of low amplitude and its existence is questionable.

The same conclusions are obtained by analyzing the Nyquist diagram in Figure 3.4. The region around the intersections between the loci is expanded in Figure 3.4b to highlight the convergent and divergent limit-cycles. The existence of the limit-cycles is confirmed by time-domain simulations. For more details read the "Shuttle Coulomb Damper" example in directory "Flixan\Examples\Payload Damper".

Bode Plot for: Outp(1)-Coulomb Damper Output Amplitud / Inpt(1)-Coulomb Damper Input Amplitude , of: DF Analysis Model (s-plane)



Nichols Plot for: Outp(1)-Coulomb Damper Output Amplitud / Inpt(1)-Coulomb Damper Input Amplitude , of: DF Analysis Model (s-plane)

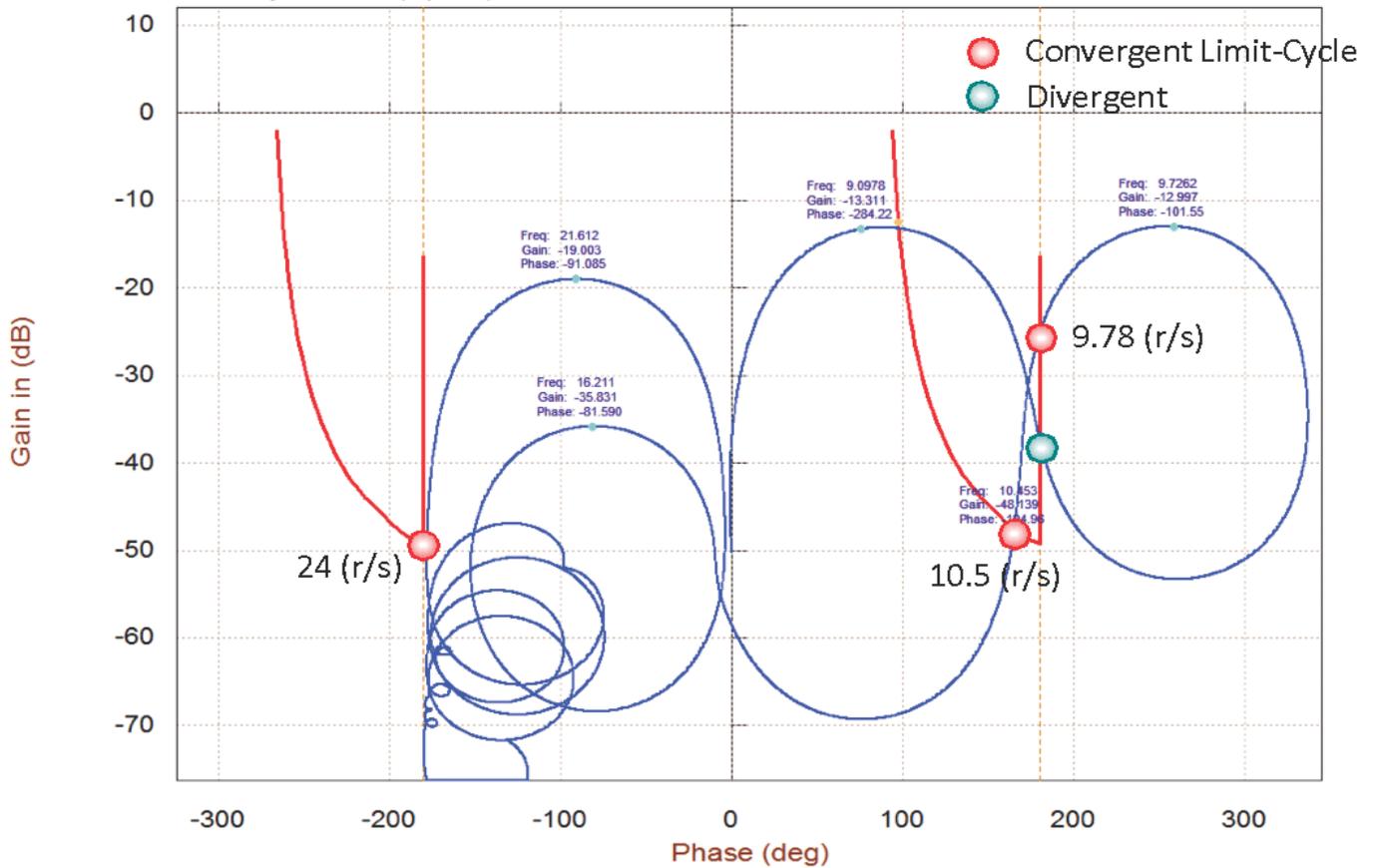
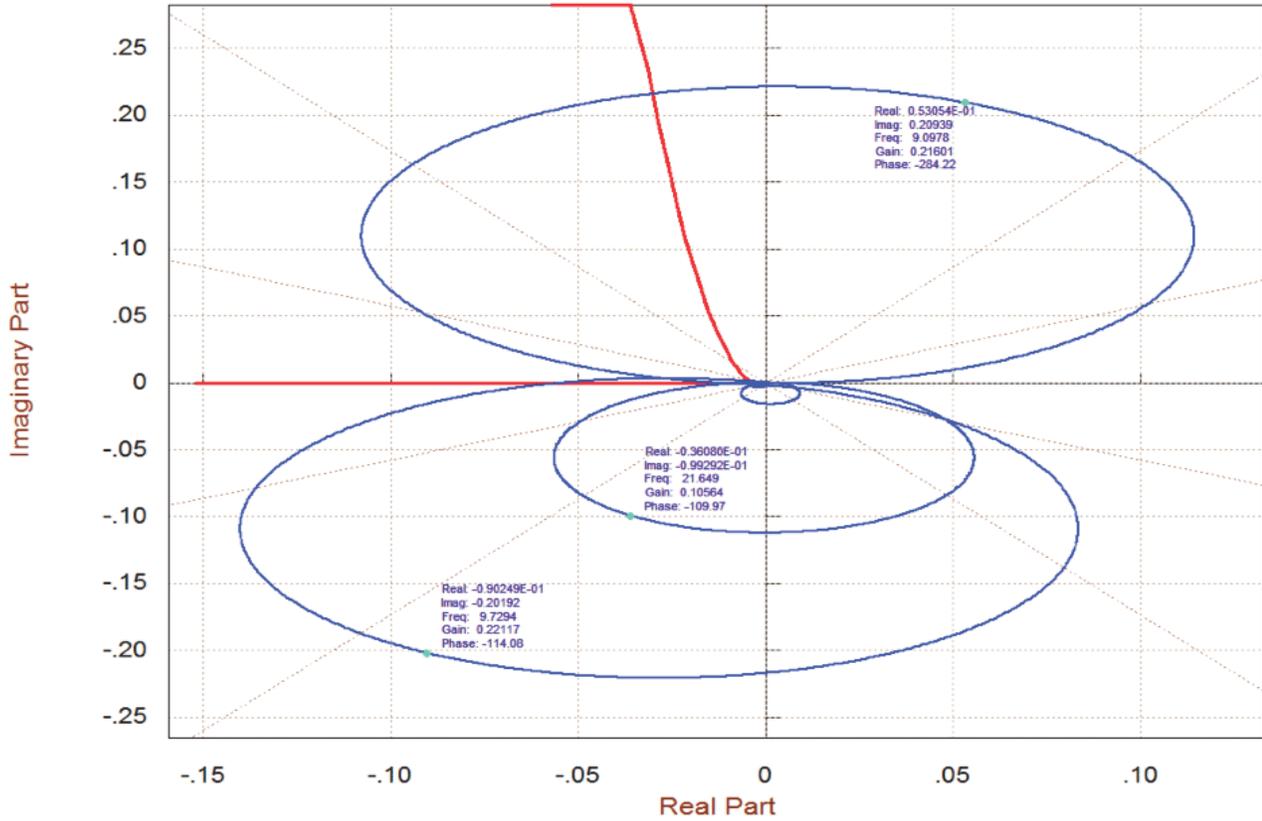


Figure 3.3 Bode and Nichols Plots across the Coulomb Damper Non-Linearity

Nyquist Plot for: Outp(1)-Coulomb Damper Output Amplitud / Inpt(1)-Coulomb Damper Input Amplitude , DF Analysis Model (s-plane)



Nyquist Plot for: Outp(1)-Coulomb Damper Output Amplitud / Inpt(1)-Coulomb Damper Input Amplitude , DF Analysis Model (s-plane)

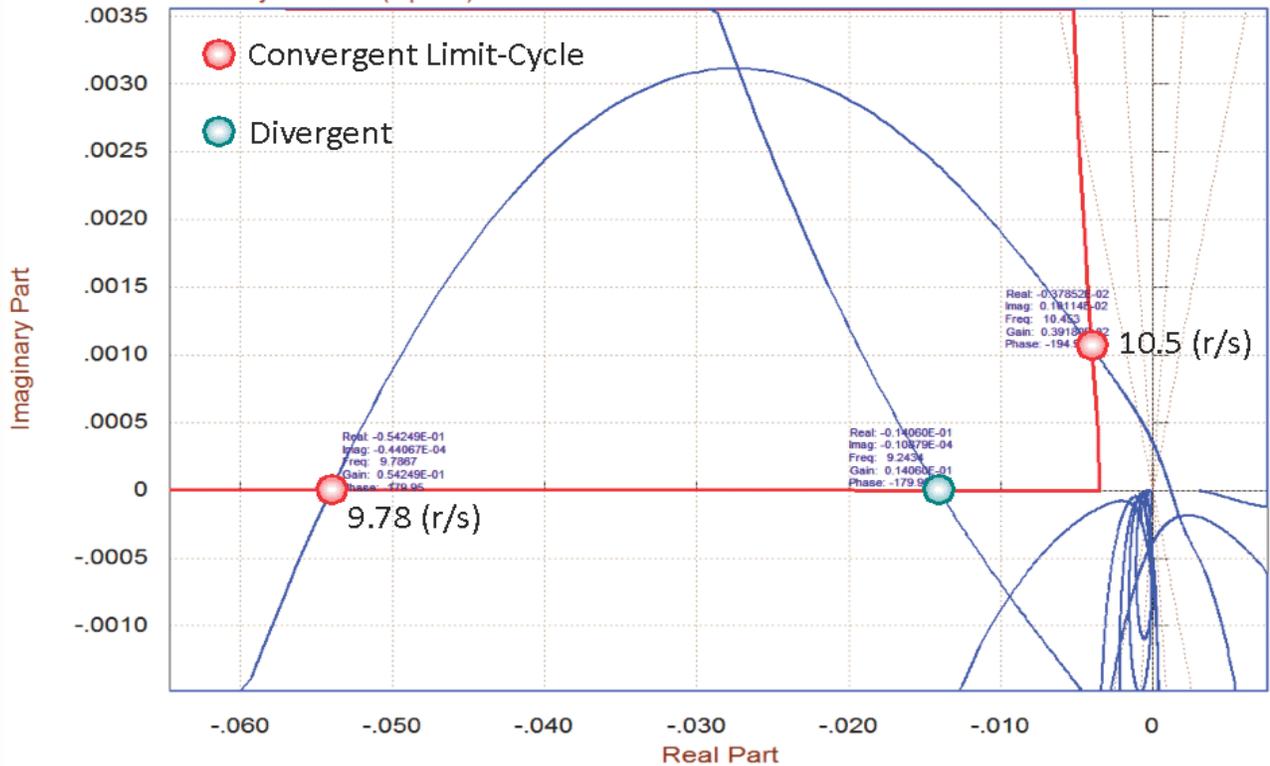


Figure 3.4 Nyquist Plot across the Coulomb Damper Non-Linearity