General: In this project, our goal is to step-by-step develop a navigation and guidance system that achieves automated landing of an autonomous unmanned aerial vehicle (drone) onto a fixed-wing aircraft that moves on a circular holding pattern. This project aims to illustrate how to design navigation techniques and navigation-based guidance techniques using linear and feedback-linearization tools.

Honor Code: You are to do your own work. Discussing the project with a friend is fine. Sharing code is not allowed.

Overview

An autonomous drone is tasked to perform environmental monitoring by collecting data such as temperature, humidity, and wind speed using on-board sensors. At some time $t_w > t_0$, the battery level drops below a known safety-critical limit. To recharge, the drone has to land on a fixed-wing aircraft that is loitering nearby. However, the communication of the drone with the fixed-wing aircraft has been lost at some time t_c, where $t_0 < t_c < t_w$. In other words, the drone can not receive information about the states of the aircraft for $t > t_c$, and the only available information to the drone about the states of the aircraft is the history of measurements in the time interval $[t_0, t_c]$. In this project, our goal is to step-by-step develop a complete navigation and guidance protocol for this drone to land on the fixed-wing aircraft using the available information from the on-board sensors so that it can charge its battery and resume its monitoring task.

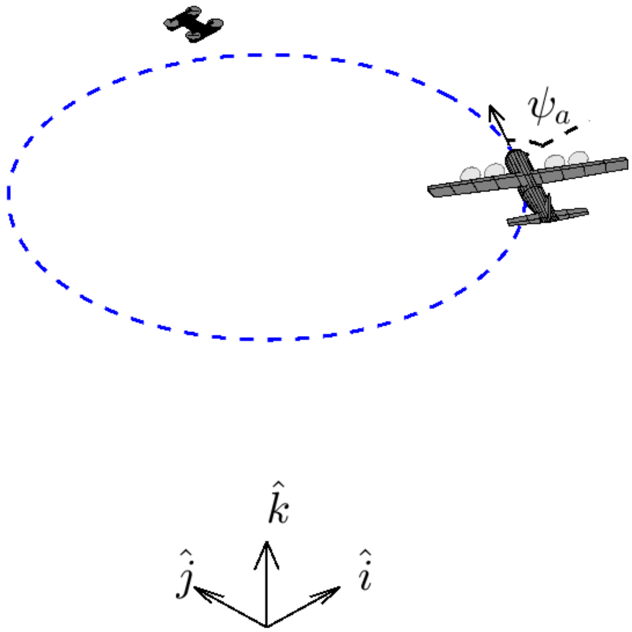
Problem 1 (25 points) *The fixed-wing aircraft is loitering at altitude $z = z_{a0}$. The equations of motion are given as:*

$$
\begin{aligned}
\dot{x}_a &= v_a \cos(\psi_a), & x_a(0) &= x_{a0}, & \text{(1a)}\\
\dot{y}_a &= v_a \sin(\psi_a), & y_a(0) &= y_{a0}, & \text{(1b)}\\
\dot{z}_a &= 0, & z_a(0) &= z_{a0}, & \text{(1c)}\\
\dot{\psi}_a &= \frac{v_a}{\rho}, & \psi_a(0) &= \psi_{a0}, & \text{(1d)}
\end{aligned}
$$

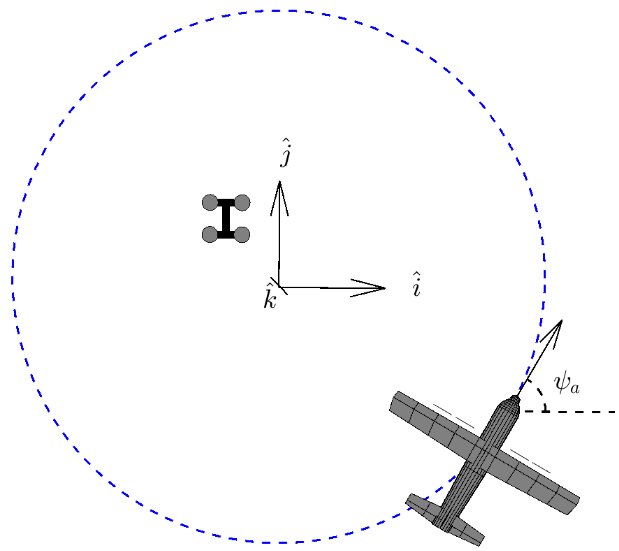
where v_a is the constant speed of the aircraft, and ρ is the constant radius of the circle on which the aircraft is moving. The drone receives the position $(x_a(t), y_a(t), z_a(t))$ and the heading $\psi_a(t)$ of the aircraft at times t that belong to the time interval $[t_0, t_c]$, i.e.,

$$\mathbf{Y}_a(t) = [x_a(t), y_a(t), z_a(t), \psi_a(t)]^T, \quad t \in [t_0, t_c]. \tag{2}$$

Let a set of sensor data obtained at N distinct time instances t_i, $i \in \{1, \dots, N\}$ be denoted as $\mathcal{H} = \Big[\mathbf{Y}_a(t_1), \quad \mathbf{Y}_a(t_2), \quad \dots \quad \mathbf{Y}_a(t_N)\Big]$.



(a) Problem Setup (3D view)



(b) Problem Setup (Top View)

(a) *(5 points) Provide closed-form expressions for the state trajectories of system* (1)*.*

(b) *(10 points) Let us denote the initial conditions in* (1) *as* q_1, q_2, q_3, q_4*, the aircraft speed as* q_5*, and the radius of the circular path as* q_6*. Given a history* $\mathcal{H}$ *of perfect measurements, provide a method to determine the parameters* $(q_1, q_2, ..., q_6)$ *of the aircraft's trajectory.*

(c) *(10 points) In practice, however, sensors provide noisy data. Assume that the measurement vector* $\mathbf{Y}_a(t_i)$*,* $i \in \{1, \ldots, N\}$*, is corrupted by a zero-mean, additive Gaussian noise vector* $\boldsymbol{\xi} = [\xi_x, \xi_y, \xi_z, \xi_\psi]^T$*, with uncorrelated components whose variances are* $\sigma^2_{\xi_x}, \sigma^2_{\xi_y}, \sigma^2_{\xi_z}, \sigma^2_{\xi_\psi}$*, respectively. Perform an error analysis for the estimates* $(\hat{q}_1, \hat{q}_2, \hat{q}_3, \hat{q}_4, \hat{q}_5, \hat{q}_6)$ *obtained from part (b). In other words, provide the expected value and the covariance matrix of the estimation error, using the history of measurements* $\mathcal{H}$*.*

Problem 2 (22 points) *You are given measurement histories of the fixed-wing aircraft under the following different scenarios along with the covariances of the respective measurement noises and the true parameters of the trajectories.*

(i) *scenario 1: aircraftData1.mat* ($\sigma^2_{\xi_x} = \sigma^2_{\xi_y} = (2.7)^2$, $\sigma^2_{\xi_z} = (1.2)^2$, $\sigma^2_{\xi_\psi} = (0.02)^2$, $N = 50$)

(ii) *scenario 2: aircraftData2.mat* ($\sigma^2_{\xi_x} = \sigma^2_{\xi_y} = (3)^2$, $\sigma^2_{\xi_z} = (2)^2$, $\sigma^2_{\xi_\psi} = (0.05)^2$, $N = 50$)

(iii) *scenario 3: aircraftData3.mat* ($\sigma^2_{\xi_x} = \sigma^2_{\xi_y} = (2.7)^2$, $\sigma^2_{\xi_z} = (1.2)^2$, $\sigma^2_{\xi_\psi} = (0.02)^2$, $N = 150$)

(a) *(12 points) For each one of the above cases, provide the estimates* $(\hat{q}_1, \hat{q}_2, ..., \hat{q}_6)$ *of the parameters* $(q_1, q_2, ..., q_6)$ *of the aircraft trajectories, expected values* $(\mu_{\tilde{q}_1}, \mu_{\tilde{q}_2}, ..., \mu_{\tilde{q}_6})$ *and the variances* $(\sigma^2_{\tilde{q}_1}, \sigma^2_{\tilde{q}_2}, ..., \sigma^2_{\tilde{q}_6})$ *of the estimation errors* $(\tilde{q}_1, \tilde{q}_2, ..., \tilde{q}_6)$ *of the parameters, where* $\tilde{q}_i = q_i - \hat{q}_i, \ \forall i \in \{1, 2, ..., 6\}$.

(b) *(4 points) Plot the x and y position coordinates of the aircraft: 1) on the actual trajectory obtained in Problem 1(a) using the true parameters provided, 2) from the measurement data, and 3) on the estimated trajectory obtained using the mean values of the parameters that you estimated in part (a), for all scenarios. For each scenario, plot the resulting 3 paths on the x-y plane in a single figure. (So, you have to provide a total of three figures in your report).*

(c) *(6 points) Compare and comment on the quality of the estimates for the three cases, i.e., compare the variances of the estimation errors of each case. Compare the plots in part (b). How does the number of measurements* N *in the history* $\mathcal{H}$ *affect the estimates?*

Note: Read the Readme.txt file for more details about the data arrangement in the MATLAB data files.

Problem 3 (10 points) *The equations of motion of the drone are given as:*

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \mathbf{u} - C_d \mathbf{v} \end{aligned} \tag{3}$$

where $\mathbf{r} = [x, y, z]^T$ *and* $\mathbf{v} = [v_x, v_y, v_z]^T$ *are the position and velocity vectors, and* $\mathbf{u}$ *is the acceleration input of the drone, all resolved in an inertial frame.* C_d *is a known drag coefficient. Position measurements that are corrupted by a zero-mean, Gaussian noise* $\boldsymbol{\epsilon}$ *with covariance matrix* $\mathbf{R}_\epsilon$ *are available:*

$$\mathbf{Y} = \mathbf{r} + \boldsymbol{\epsilon}, \tag{4}$$

where $\boldsymbol{\epsilon} = \mathcal{N}(0, \mathbf{R}_\epsilon)$. *Design a (continuous-time) Kalman filter to estimate the full state vector* $\mathbf{X} = \begin{bmatrix} \mathbf{r} \\ \mathbf{v} \end{bmatrix}$.

Problem 4 (25 points) *We now consider that the drone needs to approach the aircraft and land on its surface at location* $\mathbf{r}_l = [x_a, y_a, z_a + z_l]^T$, *where* z_l *is an offset from the aircraft's center of mass.*

(a) *(15 points) Assuming that the drone has perfect knowledge of its full state vector* $\mathbf{X}$, *design a state-feedback guidance law using the full state* $\mathbf{X}$ *and the perfect knowledge of the trajectory of the fixed-wing aircraft obtained in the Problem 1(b), so that the drone approaches the landing point* $\mathbf{r}_l$.

(b) *(10 points) We now recall that the drone has noisy measurements of its position* $\mathbf{r}$*, and that it does not directly measure its velocity* $\mathbf{v}$*. So, consider that the drone uses the estimate* $\hat{\mathbf{X}}$*, instead of the actual value of the state* $\mathbf{X}$ *in the guidance law that you designed in Problem 4(a). Provide the differential equations governing the dynamics of the guidance error* $\delta\mathbf{X} = \mathbf{X} - \begin{bmatrix} \mathbf{r}_l \\ \dot{\mathbf{r}}_l \end{bmatrix}$ *and the estimation error* $\tilde{\mathbf{X}} = \mathbf{X} - \hat{\mathbf{X}}$*.*

Hint: For part (a), you can use the idea of forcing the dynamics of the system follow a desired trajectory by choosing the control action appropriately, as was done in velocity-to-be-gained guidance. Read the chapter 7 in the main textbook (Fundamentals of Aerospace Navigation and Guidance) for more details on this.

Problem 5 (18 points) (a) *(3 points) Simulate the dynamics of the drone and the aircraft by implementing the controller that you designed in Problem 4(b) for the scenario 1 in Problem 2 with the initial condition* $\hat{\mathbf{r}}(0) = [13, 12, 112]^T m$, $\hat{\mathbf{v}}(0) = [0, 0, 0]^T m/s$*. Use the true parameters of the aircraft's trajectory, that are provided in the data files, for this sub-problem. In your report, include the plots for the error in drone's position* $\mathbf{r}$ *and the landing position* $\mathbf{r}_l = [x_a, y_a, z_a + z_l]^T$*, and for the error between drone's velocity* $\mathbf{v}$ *and landing site's velocity* $\dot{\mathbf{r}}_l = \mathbf{v}_a = [\dot{x}_a, \dot{y}_a, \dot{z}_a]^T$*. Use the following numerical data for simulation purpose:* $C_d = 0.1$, $z_l = 4\ m$,

$$\mathbf{R}_\epsilon = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.8 \end{bmatrix}, \quad \mathbf{P}_{\mathbf{r}(0)} = \begin{bmatrix} 0.9 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}, \quad \mathbf{P}_{\mathbf{v}(0)} = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}, \tag{5}$$

where $\mathbf{P}_{\mathbf{r}(0)}$ *and* $\mathbf{P}_{\mathbf{v}(0)}$ *are the covariances of* $\mathbf{r}(0)$ *and* $\mathbf{v}(0)$*. Assume* $\mathbf{r}(0)$ *and* $\mathbf{v}(0)$ *are jointly Gaussian and uncorrelated.*

(b) *(4 points) Assume that you use the steady-state Kalman gain in your Kalman filter in Problem 5(a); then, what is the steady-state covariance matrix,* $\mathbf{P}_{\delta\mathbf{X}}$*, for the guidance error* $\delta\mathbf{X}$*?*

(c) *(6 points) Now assume that the drone uses the estimated trajectory of the aircraft that you obtained using the parameters you estimated in Problem 2(a), instead of the actual aircraft trajectory information, in the feedback control law you designed. Plot the trajectories of the error* $\mathbf{r} - \mathbf{r}_l$ *and* $\mathbf{v} - \dot{\mathbf{r}}_l$ *for scenario 1 and scenario 2.*

(d) *(5 points) Is the error between the drone's position and the landing position bounded for scenario 1 in Problem 5 (c)? If yes, provide the bound on this error. If not, explain why. How does the error compare in scenario 1 and scenario 2?*

(You can use the animateTraj.m script that is provided on CANVAS for visualization.)

Michael Levy
AERO 584 Final Project
12/13/19

Table of Contents

A)

	Scenario 1		Scenario 2		Scenario 3	
Condition	$\boldsymbol{\mu}$	$\boldsymbol{\sigma^2}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma^2}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma^2}$
$q_1 = x_{a0}$	499.39	0.454	499.94	0.77	500.12	0.263
$q_2 = y_{a0}$	0.910	.344	1.27	0.47	0.32	0.22
$q_3 = y_{a0}$	0.167	0.018	0.28	0.05	0.03	$9.6 * 10^{-3}$
$q_4 = \psi_{a0}$	1.57	$9.7 * 10^{-6}$	1.57	$2.2 * 10^{-5}$	1.57	$2.5 * 10^{-6}$
$q_5 = v_a$	21.95	$8.1 * 10^{-4}$	21.94	$1.1 * 10^{-3}$	21.99	$2.0 * 10^{-4}$
$q_6 = \rho$	498.11	2.74	499.77	5.96	499.89	0.164

B)

The $\boldsymbol{P_{\delta x}} = \boldsymbol{0}_{12x12}$
Covariance matrix of the guidance error in scenario 1 is zero.

Part a:

Known:

$$\dot{x}_a = V_a \cos(\psi_a)$$
$$\dot{y}_a = V_a \sin(\psi_a)$$
$$\dot{z}_a = 0 \qquad t_0 = 0$$
$$\dot{\psi}_a = \frac{V_a}{p}$$

where v_a & p are constant w/time.

Find: Closed form expressions for the state trajectories of the system.

Solution:

$$\frac{d\psi_a}{dt} = \frac{V_a}{p}$$

$$\int_{\psi_{a0}}^{\psi_a} d\psi_a = \int_{t_0}^{t} \frac{V_a}{p}\, dt$$

$$\psi_a - \psi_{a0} = \frac{V_a}{p}(t - t_0)$$

$$\boxed{\psi_a(t) = \frac{V_a}{p} t + \psi_{a0}}$$

$$\frac{dx_a}{dt} = V_a \cos(\psi_a)$$

$$u = \psi_a = \frac{V_a}{p} t + \psi_{a0}$$

$$\int_{x_{a0}}^{x_a} dx_a = V_a \int_0^t \cos(\psi_a)\, dt \qquad \frac{du}{dt} = \frac{V_a}{p} \quad \therefore \quad dt = \frac{p}{V_a} du$$

$$x_a - x_{a0} = p \int_{u_0}^{u_t} \cos u \, du \qquad u_t = \frac{V_a}{p} t + \psi_{a0}$$

$$u_0 = \psi_{a0}$$

$$x_a = p \Big[\sin u \Big]_{u_0}^{u_t} + x_{a0}$$

$$\boxed{x_a(t) = p\left(\sin\left(\frac{V_a}{p} t + \psi_{a0}\right) - \sin(\psi_{a0})\right) + x_{a0}}$$

$$\frac{dy_a}{dt} = V_a \sin(\psi_a)$$

$$\int_{y_{a0}}^{y_a} dy_a = V_a \int_0^t \sin(\psi_a)\, dt$$

$$u = \psi_a = \frac{V_a}{\rho} t + \psi_{a0}$$

$$\frac{du}{dt} = \frac{V_a}{\rho}$$

$$dt = \frac{\rho}{V_a} du$$

$$y_a - y_{a0} = -\rho\Big[\cos u\Big]_{u_0}^{u_t}$$

$$u_t = \frac{V_a}{\rho} t + \psi_{a0} \qquad u_0 = \psi_{a0}$$

$$\boxed{y_a(t) = \rho\left(\cos(\psi_{a0}) - \cos\left(\frac{V_a}{\rho} t + \psi_{a0}\right)\right) + y_{a0}}$$

$$\frac{dz_a}{dt} = 0$$

$$\int_{z_{a0}}^{z_a} dz_a = 0$$

$$\boxed{z_a(t) = z_{a0}}$$

Part b:

Known:

$q_1 = x_{a0}$, $q_2 = y_{a0}$, $q_3 = z_{a0}$; $q_4 = \psi_{a0}$, $q_5 = V_a$, $q_6 = \rho$

History H of perfect measurements.

Find: Provide a method to determine the parameters $q_1 - q_6$.

Solution:

$$x_a(t) = q_6\left(\sin\left(\frac{q_5}{q_6} t + q_4\right) - \sin(q_4)\right) + q_1$$

$$y_a(t) = q_6\left(\cos(q_4) - \cos\left(\frac{q_5}{q_6} t + q_4\right)\right) + q_2$$

$$z_a(t) = q_3$$

$$\psi_a(t) = \frac{q_5}{q_6} t + q_4$$

In this case we have 6 unknowns and 4 equations governing the dynamics of the drone. This essentially means we have to use the "best" six initial conditions that fit the time history of our measurements. The general method that needs to be used is maximum likelihood estimation.

First I will find the jacobian that includes 4 rows that represent the derivative of each closed form expression wrt each q.

Derivatives found in MATLAB wrt each q.

$$\left.\frac{\partial g}{\partial q}\right|_0 = \begin{bmatrix} 1 & 0 & 0 & q_6(\sigma_2 - \cos(q_4)) & t\sigma_2 & \sigma_1 - \sin(q_4) - \frac{q_5 t \sigma_2}{q_6} \\ 0 & 1 & 0 & q_6(\sigma_1 - \sin(q_4)) & t\sigma_1 & \cos(q_4) - \sigma_2 - \frac{q_5 t \sigma_1}{q_6} \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{t}{q_6} & \frac{-q_5 t}{q_6^2} \end{bmatrix}$$

where $\sigma_1 = \sin\left(q_4 + \frac{q_5}{q_6}t\right)$ $\quad \sigma_2 = \cos\left(q_4 + \frac{q_5}{q_6}t\right)$

Due to the system's non-linear nature it is best to use Newton's Method to approximate the initial conditions. Since the entire measurement history is given I will use the batch method to iterate through the values of q.

Newton's Method for initial state estimation:

$$q^{K+1} = q^{K} + \left(\frac{\partial g}{\partial x}\right)^{-T}_{X^K} \left(y - g(q^{K}_{i}, t)\right)$$

6×1 $\quad$ 6×1 $\quad$ $6\times4N$ $\quad$ $4N\times1$ $\quad$ $4N\times1$

Shown above is newton's method in batch form.

$y \rightarrow$ Measurements based on aircaft data stacked for all available times.

$g(x,t) \longrightarrow$ Values of each state at evey time based on the current best guess of our initial states.

$\frac{\partial g}{\partial x} \rightarrow$ Jacobian based on our current best guess of our initial states stacked for every time value.

Then Newton's method would iterate the initial state until the change between q^K and q^{K+1} is very small!

Known: $Y_a(t_i)$ is corrupted by zero-mean, additive Gaussian noise $\mathcal{E} = [\mathcal{E}_x, \mathcal{E}_y, \mathcal{E}_z, \mathcal{E}_\psi]$ with uncorrelated components whose variances are $\sigma_{\mathcal{E}x}^2, \sigma_{\mathcal{E}y}, \sigma_{\mathcal{E}z}^2, \sigma_{\mathcal{E}\psi}^2$

Find:

Perform an error analysis for the estimates $\hat{q}_1 - \hat{q}_6$ obtained in part b. In other words, provide Expected value, covariance matrix of the estimation error, using the history of measurements H.

Solution:

We now have Knoweldge of our noise and have to generate our best guess of our initial state. The probability function $f_v(v)$ is the pdf of our noise. We want to obtain an estimate for our initial state that corresponds to higher values of $f_v(v)$. We will find $\hat{q}$ that maximizes $f_v(v)$.

$$\max f_v(v) = \max_x f_v(y - g(x))$$

which can be restated as:

$$\min_{\hat{q}} L(dx) = \frac{1}{2}(y - C\,dx)^T R_v^{-1} (dy - C\,dx)$$

where $\frac{\partial L}{\partial \hat{x}}(\hat{x}) = 0$

& the closed form solution to the first order condition.

Known → PD

$$dx = \underbrace{(C^T R_v^{-1} C)^{-1} C^T R_v^{-1}}\, dy$$

this can be used in place of our Jacobian in Newton's Method.

$$dy = (y - g(x,t)) \qquad dx = X^{K+1} - X^K$$

In order to find $\hat{q}$ we need to use Newton's Method again with a modified equation.

$$\boxed{\hat{q}^{K+1} = \hat{q}^{K} + (C^T R_v^{-1} C)^{-1} C^T R_v^{-1} (y - g(q,t))}$$

$C \rightarrow$ Jacobian Calculated in 1 b.

$$\boxed{E[\hat{q}] = E[(C^T R_v^{-1} C)^{-1} C^T R_v^{-1} (Cx + V)] = q}$$

Known that Covariance of estimation error:

$$R_{ex} = [C^T R_v^{-1} C]^{-1} C^T R_v^{-1} R_v R_v^{-1} C [C^T R_v^{-1} C]^{-1}$$

$$\boxed{R_{ex} = [C^T R_v^{-1} C]^{-1}}$$

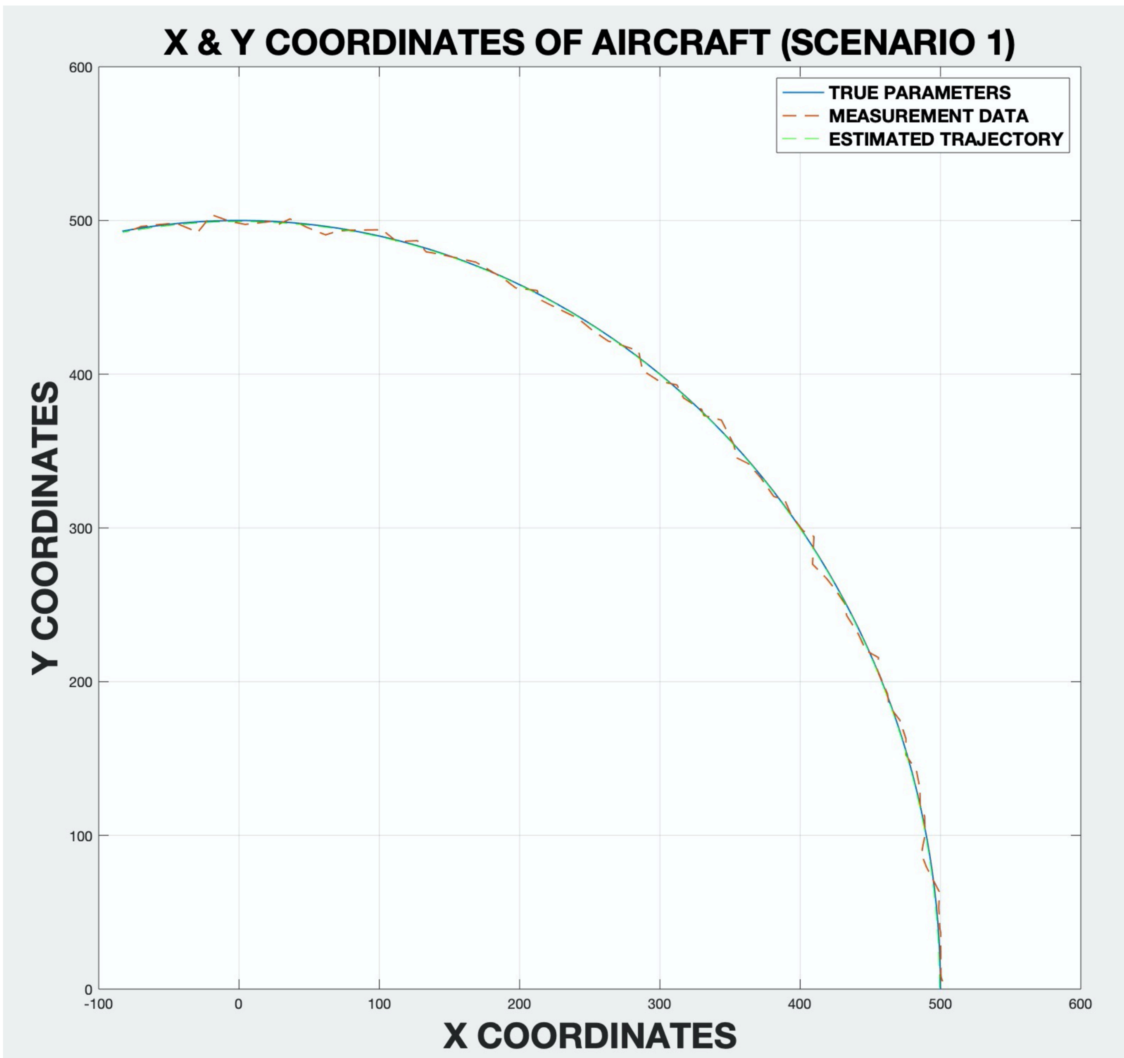
For Newton's method shown above I would use the same strategy in 1 b solving in batch with the entire time history. I would make all the matrix dimesion agree by putting R_v along a diagonal matrix with $4N \times 4N$ dimensions. The rest of the dimensions would be the same as 1b.

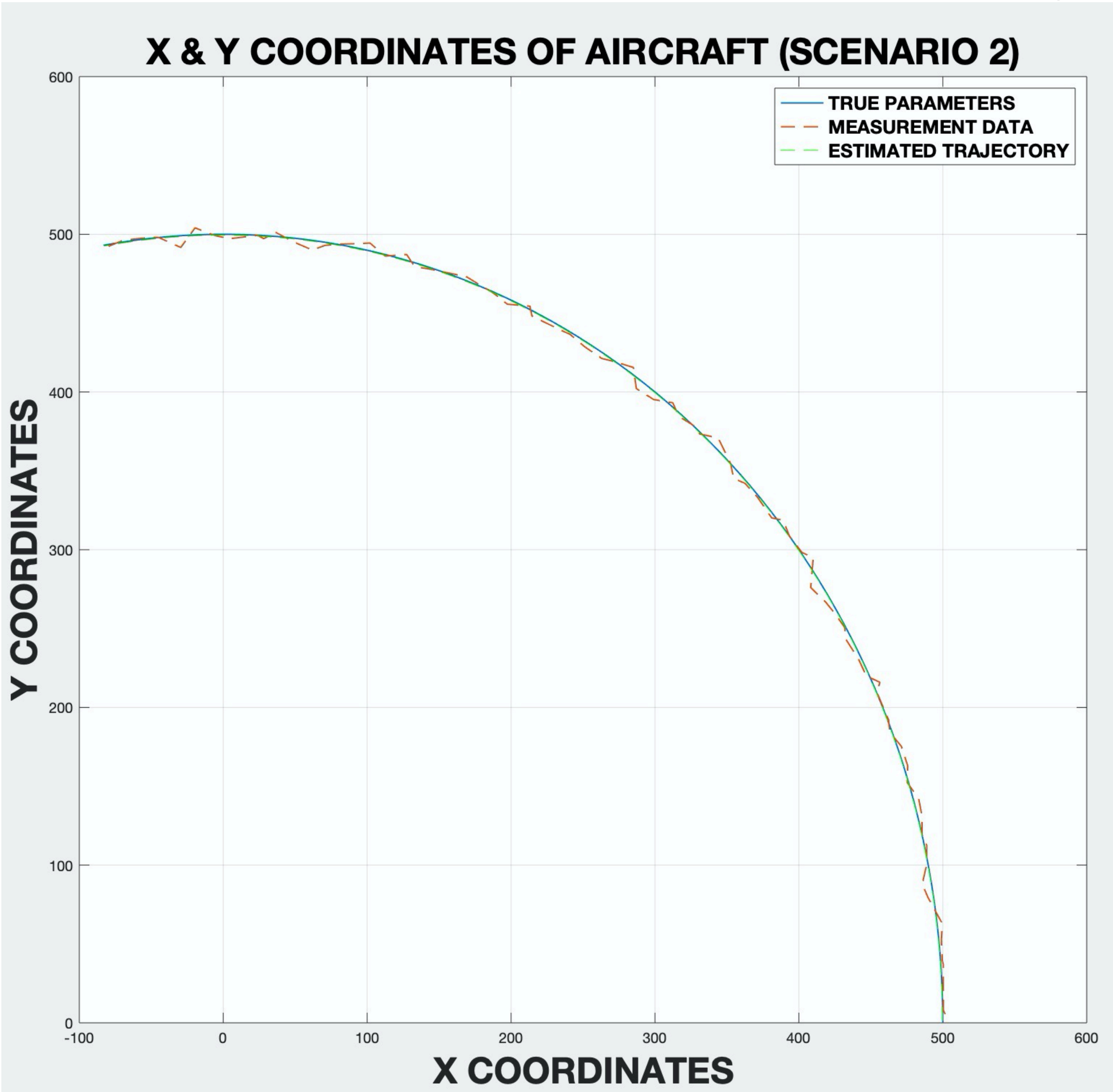
Problem 2

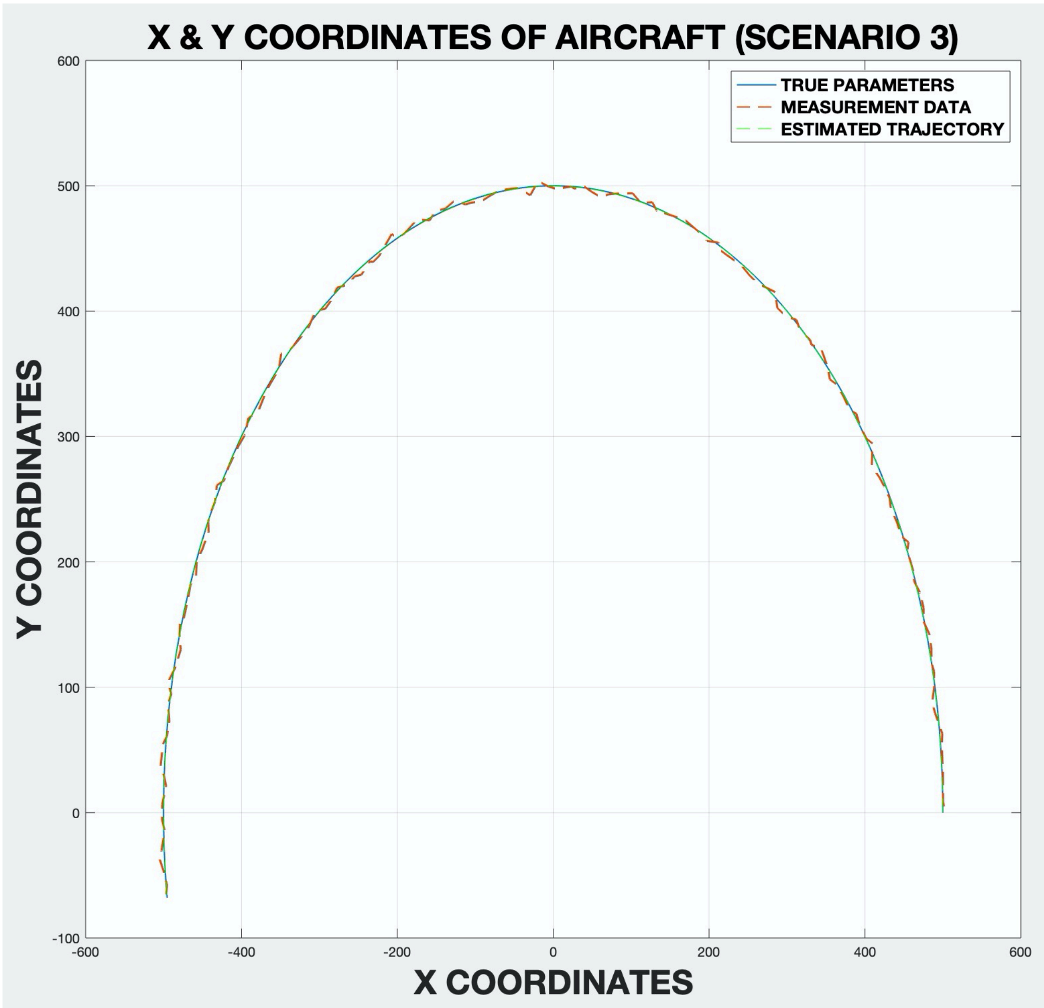
PART A)

See the above table for the rest of the values for this problem. The expected values of the estimation error as shown in problem 1 C is zero due to the fact of our covariance mean is zero.

PART B)




X & Y COORDINATES OF AIRCRAFT (SCENARIO 2)
TRUE PARAMETERS
MEASUREMENT DATA
ESTIMATED TRAJECTORY
Y COORDINATES
X COORDINATES
600
500
400
300
200
100
0
-100
0
100
200
300
400
500
600



PART C)

First off, I will compare the estimates from scenario one to scenario three. These two estimates had the same covariance matrices, however, scenario three had a larger amount of measurement samples. The variances in scenario three were always lower. This means that with a larger amount of measurements comes an increased confidence in your values. This follows along with our intuition with MLE that the more estimates you add reduces the trace of the covariance matrix.

Second, the estimates from scenario one and two show how the same number of measurements with different covariances change the user's confidence in the values found. Scenario two had larger variances in the measurements and the final values ended up having larger variances.

18

Known: where $r = [x, y, z]^T$ $v = [v_x, v_y, v_z]$

$$\dot{r} = v$$
$$\dot{v} = u - c_d v$$

all in the inertial frame. C_D is Known.

Position measurements are corrupted by a zero-mean, Gaussian noise ϵ with covariance matrix R_ϵ.

$$Y = r + \epsilon \qquad \text{where } \epsilon = \mathcal{N}(0, R_\epsilon)$$

Find: Design a (continuous-time) Kalman filter to estimate the full state vector $X = \begin{bmatrix} r \\ v \end{bmatrix}$

Solution: System:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{V}_x \\ \dot{V}_y \\ \dot{V}_z \end{bmatrix} = \begin{bmatrix} 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \\ 0&0&0&-C_D&0&0 \\ 0&0&0&0&-C_D&0 \\ 0&0&0&0&0&-C_D \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ V_x \\ V_y \\ V_z \end{bmatrix} + \begin{bmatrix} 0&0&0 \\ 0&0&0 \\ 0&0&0 \\ 1&0&0 \\ 0&1&0 \\ 0&0&1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}$$

$$\begin{bmatrix} X \\ y \\ z \\ V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ V_x \\ V_y \\ V_z \end{bmatrix} +$$

$R_v = R_\epsilon$ which is Known

Ricatti Equation:

$$\dot{P}_{\tilde{x}}(t) = A(t) P_{\tilde{x}}(t) + P_{\tilde{x}}(t) A^T(t) - P_{\tilde{x}}(t) C^T(t) R_v^{-1}(t) C(t) P_{\tilde{x}}(t) + R_w(t)$$

Where $P_{\tilde{x}} = 6 \times 6$ w/ 36 elements

Solve the ricatti equation w/SS Solution $\dot{P}_{\tilde{x}}(t)=0$

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} O_3 & I_3 \\ O_3 & -C_D I_3 \end{bmatrix}\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} + \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}\begin{bmatrix} O_3 & O_3 \\ I_3 & -C_D I_3 \end{bmatrix}$$

$$-\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}\begin{bmatrix} I_3 \\ O_3 \end{bmatrix} \cdot R_v^{-1} \cdot \begin{bmatrix} I_3 & O_3 \end{bmatrix}\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} + 0$$

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} P_{21} & P_{22} \\ -C_D P_{21} & -C_D P_{22} \end{bmatrix} + \begin{bmatrix} P_{12} & -C_D P_{12} \\ P_{22} & -C_D P_{22} \end{bmatrix}$$

$$-\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}\begin{bmatrix} I_3 \\ O_3 \end{bmatrix} R_v^{-1} \begin{bmatrix} P_{11} & P_{12} \end{bmatrix}.$$

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} P_{21} + P_{12} & P_{22} - C_D P_{12} \\ P_{22} - C_D P_{21} & -2C_D P_{22} \end{bmatrix} + \begin{bmatrix} P_{11} \\ P_{21} \end{bmatrix} R_v^{-1} \begin{bmatrix} P_{11} & P_{12} \end{bmatrix}$$

Note: Each element of $P_{\hat{x}}$ is a 3×3 matrix

$$P_{\hat{x}} = \begin{bmatrix} P_{11} & P_{21} & P_{31} & P_{41} & P_{51} & P_{61} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{41} & P_{24} & P_{34} & P_{44} & P_{45} & P_{46} \\ P_{51} & P_{25} & P_{35} & P_{45} & P_{55} & P_{56} \\ P_{61} & P_{26} & P_{36} & P_{46} & P_{56} & P_{66} \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}$$

$P_{12} = P_{21}$ due to symmetry

Due to symmetry of P_{12}, P_{11}, and P_{22} unknowns I end up with 21 equations, and 21 unknowns.

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 2P_{12} + R_v^{-1} P_{11}^2 & P_{22} - C_D P_{12} + R_v^{-1} P_{11} \cdot P_{12} \\ P_{22} - C_D P_{12} + R_v^{-1} \cdot P_{11} \cdot P_{12} & R_v^{-1} P_{12}^2 - 2C_D P_{22} \end{bmatrix}$$

Next, after solving for all of the values for the steady state $P_{\tilde{x}}$, find the optimal Kalman gain.

Optimal Kalman Gain:

$$G(t) = P_{\tilde{x}} C^T R_v^{-1}$$

Substitute Into Estimator:

$$\boxed{\dot{\hat{x}} = A(t)\hat{x}(t) + B(t)u(t) + G(t)(C(t)\hat{x}(t) - y(t))}$$

where $\hat{x}$ is our estimate of our states.

Known:

$r_\ell = [x_a, y_a, z_a + Z_\ell]^T$

Drone has perfect knowledge of its full state vector X.
Perfect Knowledge of the trajectory of the fixed aircraft from 1(b).

Find: Design a state-feedback guidance law using the full state X.

Solution: $\delta x = x(t) - x^o(t)$ $\delta u = u(t) - u^o(t)$

where $x^o(t)$ & $u^o(t)$ are from nominal trajectory.

$$\begin{bmatrix} \dot{x}_o \\ \dot{y}_o \\ \dot{z}_o \\ \dot{V}_{xo} \\ \dot{V}_{yo} \\ \dot{V}_{zo} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -C_D & 0 & 0 \\ 0 & 0 & 0 & 0 & -C_D & 0 \\ 0 & 0 & 0 & 0 & 0 & -C_D \end{bmatrix} \begin{bmatrix} x_o \\ y_o \\ z_o \\ V_{xo} \\ V_{yo} \\ V_{zo} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{xo} \\ u_{yo} \\ u_{zo} \end{bmatrix}$$

nominal trajectory is plane trajectory where:

$$\dot{x}_o = V_a \cos\left(\frac{V_a}{p}t + \psi_{ao}\right)$$

$$\dot{y}_o = V_a \sin\left(\frac{V_a}{p}t + \psi_{ao}\right)$$

$$\dot{z}_o = 0$$

$$\dot{V}_{xo} = \frac{-V_a^2}{p} \sin\left(\frac{V_a}{p}t + \psi_{ao}\right)$$

$$\dot{V}_{yo} = \frac{V_a^2}{p} \cos\left(\frac{V_a}{p}t + \psi_{ao}\right)$$

$$\dot{V}_{zo} = 0$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ u_{xo} \\ u_{yo} \\ u_{zo} \end{bmatrix} = \begin{bmatrix} V_{xo} \\ V_{yo} \\ V_{zo} \\ \dot{V}_{xo} \\ \dot{V}_{yo} \\ \dot{V}_{zo} \end{bmatrix} - \begin{bmatrix} V_{xo} \\ V_{yo} \\ V_{zo} \\ -C_D V_{xo} \\ -C_D V_{yo} \\ -C_D V_{zo} \end{bmatrix}$$

$$u_{xo} = \dot{V}_{xo} + C_D X_o$$

$$u_{xo} = \frac{-V_a^2}{P} \sin\left(\frac{V_a}{P} t + \psi_{ao}\right) + C_D V_a \cos\left(\frac{V_a}{P} t + \psi_{ao}\right)$$

$$u_{yo} = \frac{V_a^2}{P} \cos\left(\frac{V_a}{P} t + \psi_{ao}\right) + C_D V_a \sin\left(\frac{V_a}{P} t + \psi_{ao}\right)$$

$$u_{zo} = 0$$

$$d\dot{x} = A(t)\, dx + B(t)\, du$$

$$d\dot{x} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -C_D & 0 & 0 \\ 0 & 0 & 0 & 0 & -C_D & 0 \\ 0 & 0 & 0 & 0 & 0 & -C_D \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \\ dV_x \\ dV_y \\ dV_z \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} u_x$$

State x is Known perfectly in flight ∴ dx is perfectly measured in flight. du needs to be of the form so that it drives dx to zero.

∴ $\boxed{du = F(t)\, dx}$

F(t) is only unknown. We want $\lim_{t \to \infty} dx(t) = 0$

$$\therefore\ d\dot{x} = \left(\underbrace{\begin{bmatrix} \vec{O}_3 & \vec{I}_3 \\ \vec{O}_3 & -C_D \vec{I}_3 \end{bmatrix}}_{A} + \underbrace{\begin{bmatrix} \vec{O}_3 \\ I_3 \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} f_1 & f_2 & f_3 & f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 & f_{10} & f_{11} & f_{12} \\ f_{13} & f_{14} & f_{15} & f_{16} & f_{17} & f_{18} \end{bmatrix}}_{F} \right) dx$$

From the $d\dot{x}$ equation we can use the MATLAB place function to find values for our F matrix that will drive $dx \rightarrow$ as time appraches ∞. This follows a similar procedure as defining gains that make a system observable, and F ~~ga~~ gains that make a system controllable!

Now that we Know $\rightarrow du = F dx$ (Guidance Law)

$$u - u_0 = F(x - x_0)$$

$$u = F(x - x_0) + u_0$$

x_0 is Known as the plane's trajectory

u_0 was solved for

F is Known from calculations

X we have perfect Knoweldge of.

Therefore, our thrust input for our drone is the only unKnown.

Checking Controllability of our LTI system:

$$C = [B \;\; AB \;\; A^2B \;\; A^3B \;\; A^4B \;\; A^5B]$$

Using MATLAB $\text{rank}(C) = 6 \therefore$ full rank this system designed is controllable.

Known: Drone has noisy measurents of position r.
Drone uses estimate $\hat{x}$.

Find: Differential Equations governing the dynamics of the guidance error $\delta X = X - \begin{bmatrix} r_1 \\ \dot{r}_2 \end{bmatrix}$ & est. error $\tilde{x} = x - \hat{x}$

Solution:

W/ corruption by navigation errors our system is:

$$d\dot{x}(t) = A(t)\,dx(t) + B(t)\,du(t)$$

$$\delta y(t) = C(t)\,dx(t) + v(t)$$

Combined navigation & guidance linear navigator

$$d\dot{\hat{x}}(t) = A(t)\,d\hat{x}(t) + B(t)\,du(t) + G(t)\big(C(t)\,d\hat{x}(t) - dy(t)\big)$$

$$\& \quad du(t) = F(t)\,d\hat{x}(t)$$

Estimation Error $= d\tilde{x} = dx - d\hat{x}$

Closed-loop dynamics of the actual systems under the guidance law $du(t) = F(t)\,d\hat{x}(t)$

$$d\dot{x}(t) = A(t)\,dx(t) + B(t)F(t)\,dx(t) - B(t)F(t)\,d\tilde{x}(t) + w(t)$$

$$d\dot{\tilde{x}}(t) = A(t)\,d\tilde{x} + G(t)C(t)\,d\hat{x}(t) + G(t)\,v(t) + w(t)$$

In matrix form:

guidance

navigation error!

$$\boxed{\begin{bmatrix} d\dot{x} \\ d\dot{\tilde{x}} \end{bmatrix} = \begin{bmatrix} A(t) + B(t)F(t) & -B(t)F(t) \\ 0 & A(t) + G(t)C(t) \end{bmatrix} \begin{bmatrix} dx \\ d\tilde{x} \end{bmatrix} + \begin{bmatrix} I & 0 \\ I & G(t) \end{bmatrix} \begin{bmatrix} w(t) \\ v(t) \end{bmatrix}}$$

Defining the matrices that make up our guidance error & estimation error differential equations.

$$A = \begin{bmatrix} 0_3 & I_3 \\ 0_3 & -C_D I_3 \end{bmatrix} \quad B = \begin{bmatrix} 0_3 \\ I_3 \end{bmatrix} \quad C = \begin{bmatrix} I_3 & 0_3 \end{bmatrix}$$

$F = 3\times6$ of individual gains $f_1 \rightarrow f_{18}$

$G = 6\times3$ of individual gains $g_1 \rightarrow g_{18}$

$w(t)$ is zero in our case

$v(t)$ is included in our model.

Known: $\hat{r}(0)$, $\hat{V}(0)$, r_ℓ, C_D, z_ℓ, R_ε, $P_{r(0)}$, $P_{V(0)}$

r(0) & V(0) are jointly Gaussian & uncorrelated.

Solution:

$$A = I_6 + AT = \begin{bmatrix} 1 & 0 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 & T \\ 0 & 0 & 0 & 1-C_DT & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-C_DT & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-C_DT \end{bmatrix}$$

Find discretized A & B matrices.

$$B = BT = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ T & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & T \end{bmatrix} \qquad y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ V_x \\ V_y \\ V_z \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \end{bmatrix}$$

Discrete Kalman Filter steps:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \quad \text{(Guess State)}$$

$$P_k^- = AP_{k-1}A^T + \cancel{R_{w,k-1}}^{\,0} \quad \text{(Guess Covariance)}$$

$$K_k = P_k^- C_k^T (C_k P_k^- C_k^T + R_v)^{-1} \quad \text{(Kalman Gain)}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C_k\hat{x}_k^-) \quad \text{(Estimate w/ measurement)}$$

$$P_k = (I - K_kC_k)P_k^-$$

y_k is calculated w/ the closed form equations and then randomized w/ zero mean Gaussian noise with covariance from our initial measurements.

During each time step I will recalculate the control input $u = F\,dx + u_0$ (guidance law from 4a

<u>Known:</u> Assume you use the steady-state Kalman gain in your Kalman Gain in problem 5a.

<u>Find:</u> What is the SS covariance matrix, $P_{\delta x}$, for the guidance error δX?

<u>Solution:</u>

Assuming that our initial conditions are Gaussian, and $W(t)$ / $V(t)$ are zero-mean, Gaussian, white & uncorrelated

From the combined matrix form of the guidance law and navigation error equations can be re-written

$$\dot{\xi} = A(t)\,\xi + \Gamma(t)\,X \qquad \& \quad R_x = \begin{bmatrix} R_W & 0 \\ 0 & R_V \end{bmatrix}$$

w/covariance of ξ is $P_\xi(t)$

<u>Lyapunov Equation:</u>

$$\dot{P}_\xi(t) = A(t)\,P_\xi(t) + P_\xi(t)\,A^T(t) + \Gamma(t)\,R_x(t)\,\Gamma^T(t)$$

$$SS \therefore \dot{P}_\xi(t) = 0$$

$$0 = \begin{bmatrix} A+BF & -B\cdot F \\ 0 & A+GC \end{bmatrix} \begin{bmatrix} P_{\xi 11} & P_{\xi 12} \\ P_{\xi 21} & P_{\xi 22} \end{bmatrix} + \begin{bmatrix} P_{\xi 11} & P_{\xi 12} \\ P_{\xi 21} & P_{\xi 22} \end{bmatrix} \begin{bmatrix} A+BF & 0 \\ -BF & A+GC \end{bmatrix}$$

$$+ \begin{bmatrix} I & 0 \\ 0 & G \end{bmatrix} \begin{bmatrix} R_W & 0 \\ 0 & R_V \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & G \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} (A+BF)P_{\xi 11} - BF P_{\xi 21} & (A+BF)P_{\xi 12} - BF P_{\xi 22} \\ (A+GC)P_{\xi 21} & (A+GC)P_{\xi 22} \end{bmatrix}$$

$$+ \begin{bmatrix} P_{\xi 11}(A+BF) - P_{\xi 12}BF & P_{\xi 12}(A+GC) \\ P_{\xi 21}(A+BF) - P_{\xi 22}BF & P_{\xi 22}(A+GC) \end{bmatrix}$$

$$+ \begin{bmatrix} R_w & 0 \\ 0 & GR_vG \end{bmatrix}$$

From the form of A & the continuous time Kalman filter gain equation: $G = -P_{\xi 22} C^T R_v^{-1}$

<u>Equation 4:</u>

$$0 = (A+GC)P_{\xi 22} + P_{\xi 22}(A+GC) + GR_vG$$

$$0 = AP_{\xi 22} + GCP_{\xi 22} + P_{\xi 22}A + P_{\xi 22}GC + GR_vG$$

$$0 = AP_{\xi 22} - P_{22}C^TR_v^{-1}CP_{\xi 22} + P_{\xi 22}A - P_{\xi 22} \cdot P_{\xi 22}C^TR_v^{-1}C + P_{\xi 22}C^TP_{\xi 22}C^TR_v^{-1}$$

Equation is in terms of A, C, R_v, and $P_{\xi 22}$

we Know A, C, and R_v are + real numbers therefore the solution for $P_{\xi 22}$ must be that $P_{\xi 22} = 0$.

<u>Equation 2:</u>

$$0 = (A+BF)P_{\xi 12} - BF\cancelto{0}{P_{\xi 22}} + P_{\xi 12}(A + \cancelto{0}{GC})$$

$$0 = (A+BF)P_{\xi 12} + P_{\xi 12}A$$

A, B, & F are all + $\therefore$ $P_{\xi 12} = 0$

<u>Equation 3:</u>

$$0 = (A + \cancel{GC}^{\,0}) P_{\xi 21} + P_{\xi 21}(A + BF) - \cancel{P_{\xi 22}}^{\,0} BF$$

A, B, & F are all $\neq$ $\therefore$ $P_{\xi 21} = 0$

<u>Equation 1:</u>

$$0 = (A + BF) P_{\xi 11} - BF\cancel{P_{\xi 21}}^{\,0} + P_{\xi 11}(A + BF) - \cancel{P_{\xi 12}}^{\,0} BF + \cancel{R_w}^{\,0}$$

$$0 = (A + BF) P_{\xi 11} + P_{\xi 11}(A + BF)$$

A, B, & F are all + $\therefore$ $P_{\xi 11} = 0$

P_ξ is a 12×12 of all zeros

$$\boxed{P_\xi = 0_{12 \times 12}}$$

Find: Is there error between the drone's position & the landing position bounded for scenario 1 in problem 5(c)? If yes provide the bound for this error. How does error compare in scenario 1 & 2?

Solution:

Taking our estimates for our state w/ the closed form equations in part 1 b we get:

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{V}_x \\ \hat{V}_y \\ \hat{V}_z \end{bmatrix} = \begin{bmatrix} \hat{q}_6\left(\sin\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right) - \sin(\hat{q}_4)\right) + \hat{q}_1 \\ \hat{q}_6\left(\cos(\hat{q}_4) - \cos\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right)\right) + \hat{q}_2 \\ \hat{q}_3 \\ \hat{q}_5 \cos\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right) \\ \hat{q}_5 \sin\left(\hat{q}_5/\hat{q}_6 t + \hat{q}_4\right) \\ 0 \end{bmatrix}$$

$\begin{bmatrix} x \\ y \\ z \\ V_x \\ V_y \\ V_z \end{bmatrix} \rightarrow$ use the same governing equations except with the true initial values!

$$x - \hat{x} = q_6\left(\sin\left(\frac{q_5}{q_6}t + q_4\right) - \sin(q_4)\right) + q_1$$
$$- \hat{q}_6\left(\sin\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right) - \sin(\hat{q}_4)\right) - \hat{q}_1$$

x & $\hat{x}$ are both described by sinusodial equations w/ q_5/q_6 acting as the frequency and q_4 acting as the phase shift.

For Scenario 2:

$q_5 \neq \hat{q}_5$, $q_6 \neq \hat{q}_6$, $q_4 \neq \hat{q}_4$

The sine terms that contain q & $\hat{q}$ have different frequencies and phase shifts due to the relationships above. At some point in the the sine terms with $\hat{q}$ & q will equal 1 & -1. This is where the maximum bounded error occurs!

$$|x-\hat{x}| = q_6 \sin\left(\frac{q_5}{q_6}t + q_4\right) - \hat{q}_6 \sin\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right)$$

$$\hat{q}_6 \sin(\hat{q}_4) - q_6 \sin(q_4) + q_1 - \hat{q}_1$$

$$|x-\hat{x}| = 500 \cdot 1 - 499.8 \cdot -1 + 499.8 \cdot \sin(1.57)$$
$$- 500 \cdot \sin(1.571) + 500 - 499.94$$

$$\boxed{|x-\hat{x}| = 999.7\text{m max bounded error}}$$

This relationship holds true for y error as well.

$$|y-\hat{y}| = -q_6 \cos\left(\frac{q_5}{q_6}t + q_4\right) + \hat{q}_6 \cos\left(\frac{\hat{q}_5}{\hat{q}_6}t + \hat{q}_4\right)$$
$$- \hat{q}_6 \cos(\hat{q}_4) + q_6 \cos(q_4) + q_2 - \hat{q}_2$$

$$|y-\hat{y}| = -500 \cdot 1 + 499.7 \cdot 1 - 499.8 \cdot \cos(1.574)$$
$$+ 500 \cdot \cos(1.570) + 0 - 1.267$$

$$\boxed{|y-\hat{y}| = 998.5\text{ m max bounded error}}$$

$$|z - \hat{z}| = q_3 - \hat{q}_3 = 0 - 0.279$$

$$\boxed{|z - \hat{z}| = 0.279 \quad \text{max } z \text{ error}}$$

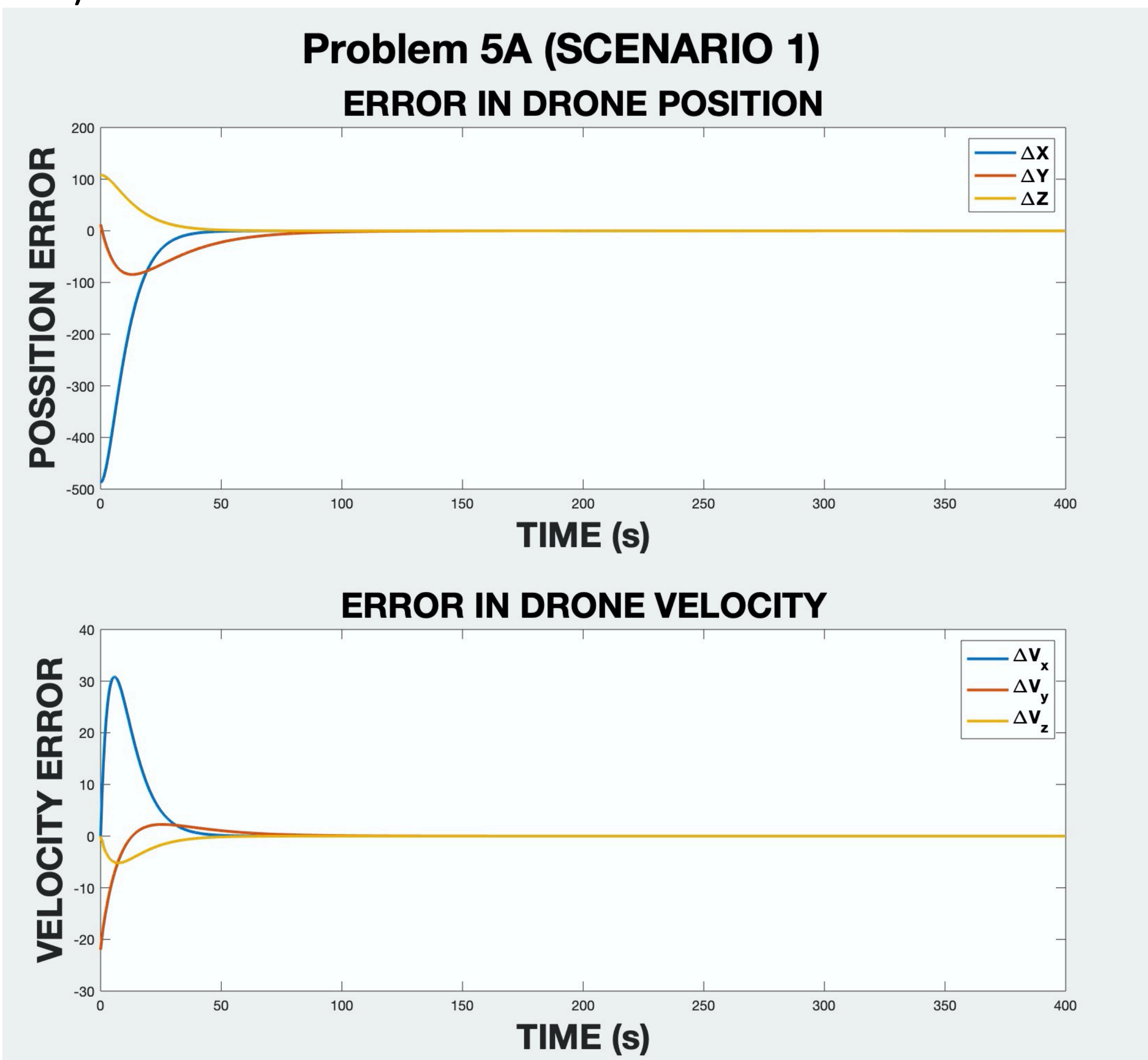
For Scenario 1:

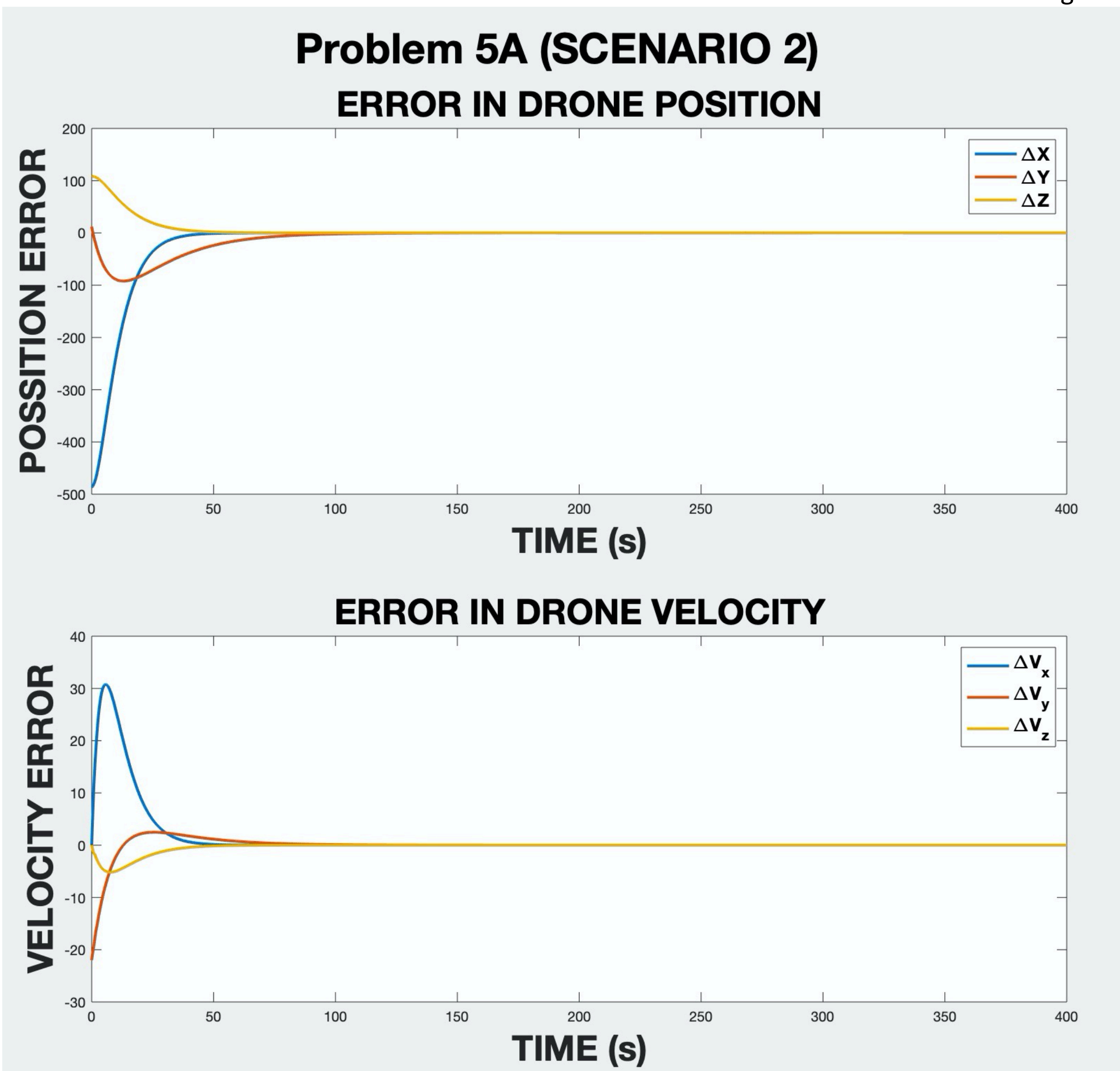
$$|x - \hat{x}| = 996.8 \text{ m} \quad \text{max error}$$
$$|y - \hat{y}| = 999.1 \text{ m} \quad \text{max error}$$
$$|z - \hat{z}| = 0.167 \text{ m} \quad \text{max error}$$

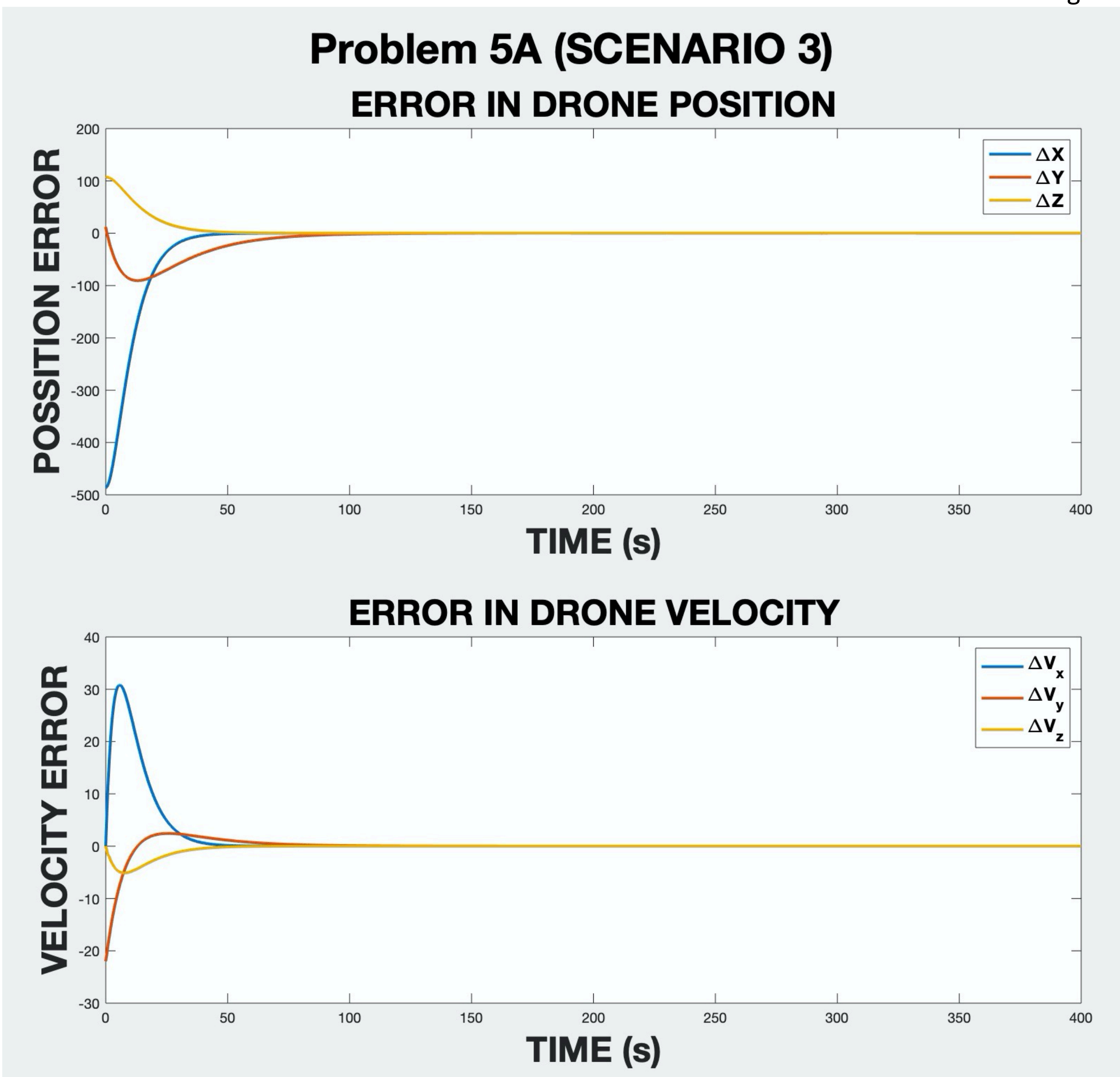
Both of these systems exhibit roughly the same max errors due to having a phase shift and different frequencies, which causes build up a certain times as t -> oo!

Problem 5

PART A)



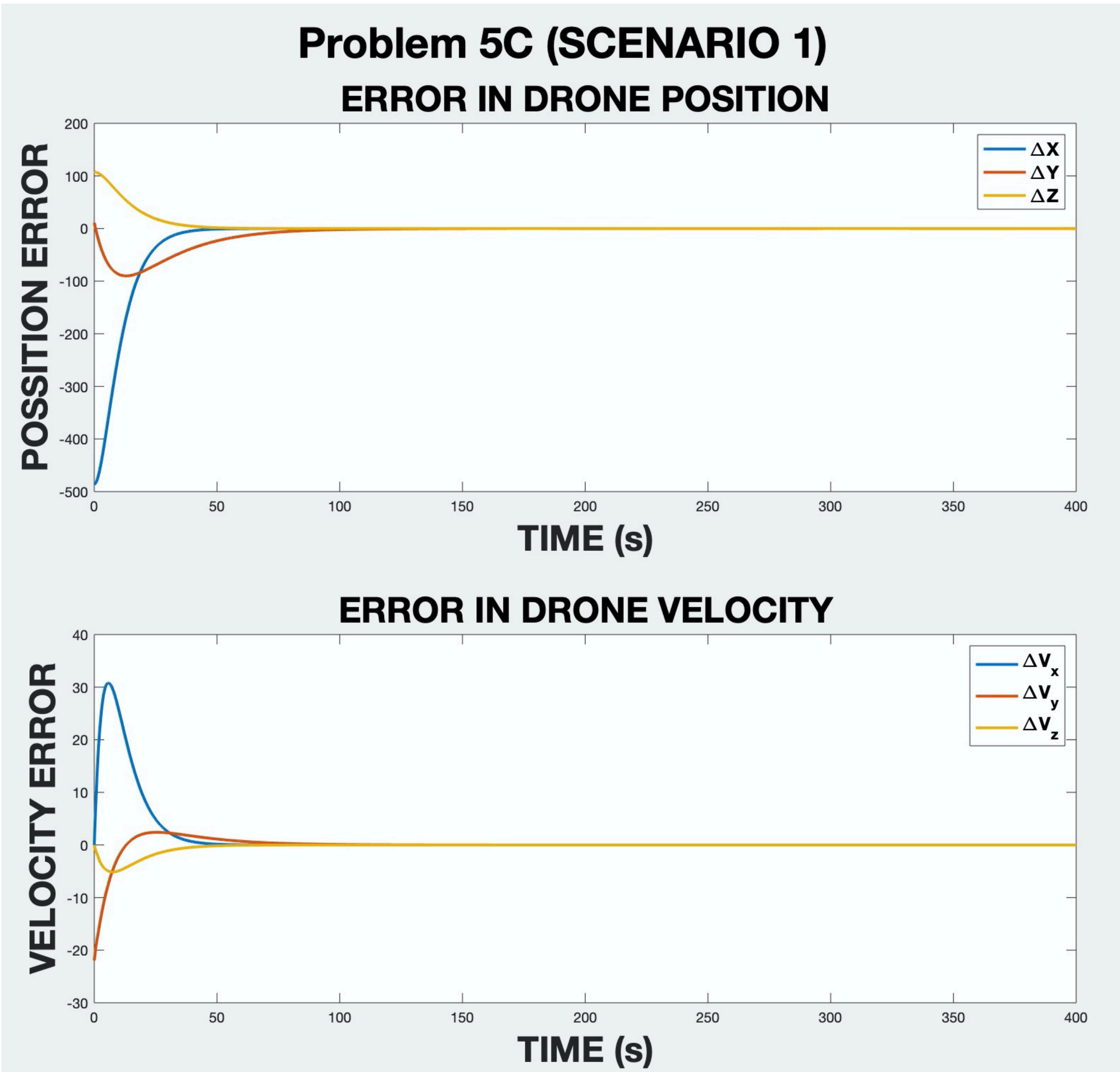

Problem 5A (SCENARIO 2)
ERROR IN DRONE POSITION
POSSITION ERROR
TIME (s)
ΔX
ΔY
ΔZ
200
100
0
-100
-200
-300
-400
-500
0
50
100
150
200
250
300
350
400
ERROR IN DRONE VELOCITY
VELOCITY ERROR
TIME (s)
ΔVx
ΔVy
ΔVz
40
30
20
10
0
-10
-20
-30
0
50
100
150
200
250
300
350
400

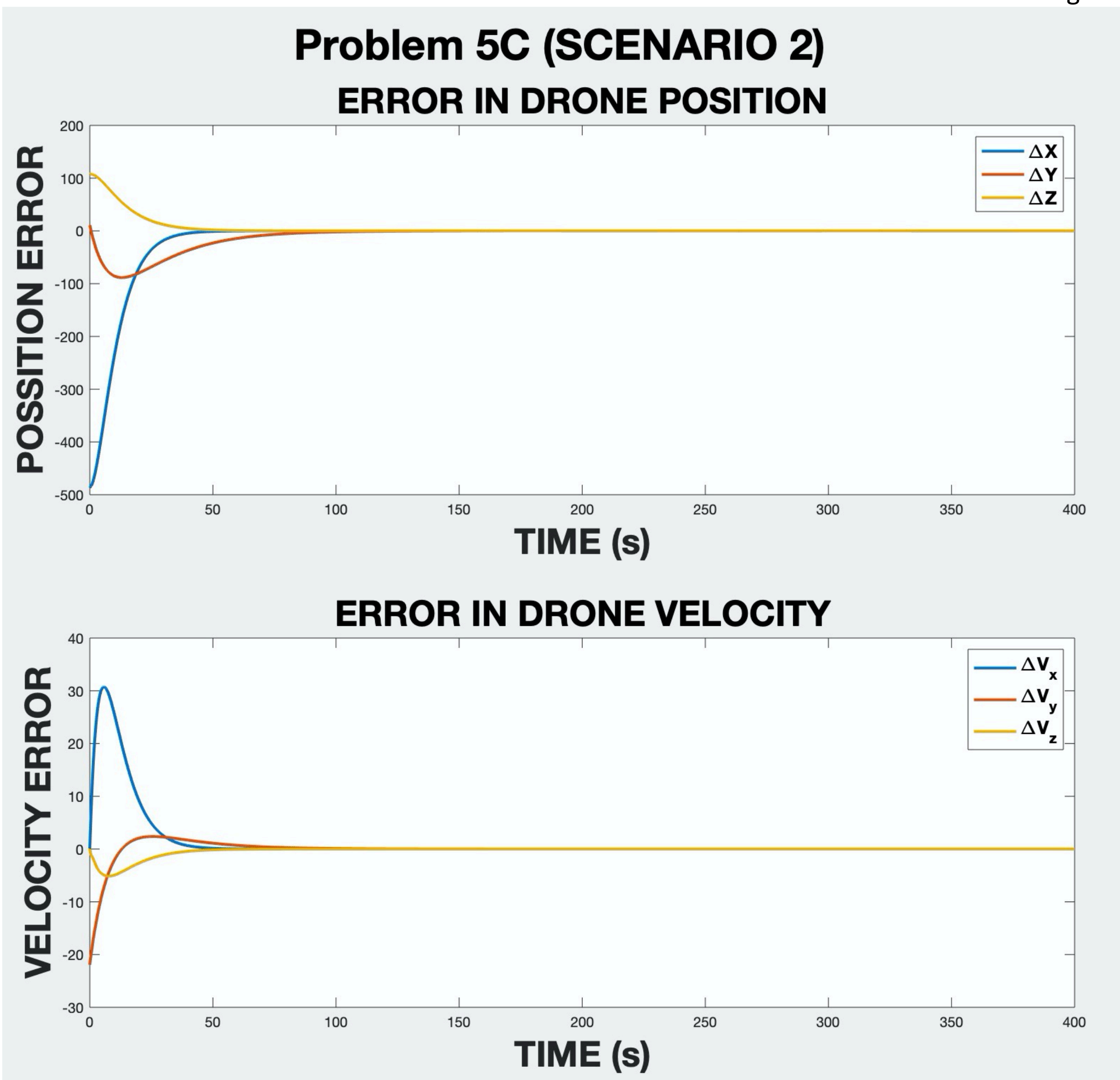

Problem 5A (SCENARIO 3)
ERROR IN DRONE POSITION
POSSITION ERROR
TIME (s)
ΔX
ΔY
ΔZ
ERROR IN DRONE VELOCITY
VELOCITY ERROR
TIME (s)
ΔV_x
ΔV_y
ΔV_z

Problem 5

PART C)

In part C I used the estimate of the plane's initial states in my Kalman filter. I then found the difference between the position of the drone and then true trajectory of the plane using the q* estimates. Below are Scenario 1 and 2.




Problem 5C (SCENARIO 2)
ERROR IN DRONE POSITION
POSSITION ERROR
TIME (s)
ΔX
ΔY
ΔZ
200
100
0
-100
-200
-300
-400
-500
0
50
100
150
200
250
300
350
400
ERROR IN DRONE VELOCITY
VELOCITY ERROR
TIME (s)
ΔVx
ΔVy
ΔVz
40
30
20
10
0
-10
-20
-30
0
50
100
150
200
250
300
350
400

Problem 1

```
clc
clear
clf
close all
load("aircraftData1.mat");
set(groot,'defaultfigureposition',[600 300 900 900]);

% % THIS CODE WAS USED FOR PROBLEM 1

% Initial Values
N = length(aircraftData.t);

% Initial Guess +
q_old = [470; 0.5; 0.2; 1.1; 24; 520];

% Pre-allocate
jacob = zeros(4 * N, 6);
y = zeros(4 * N, 1);
g = zeros(4 * N, 1);
Rv_matrix = zeros(4 * N, 4 * N);
flg = 0;

% MLE To calculate initial states
while(flg == 0)
    for i = 1:N
        jacob(4 * i - 3 : 4 * i, :) = jacobian1(q_old.', aircraftData.t(i));
        y(4 * i - 3 : 4 * i, 1) = aircraftData.Ya(:,i);
        g(4 * i - 3 : 4 * i, 1) = states(q_old.', aircraftData.t(i));
    end
    q_new = q_old + pinv(jacob) * (y - g);

    if abs(q_new - q_old) < .00001
        flg = 1;
    end
    q_old = q_new;
end



% Part 2b

% Calculate Aircraft Parameters with Known initial states
states1 = zeros(N,4);
states2 = zeros(N, 4);
states3 = zeros(N, 4);
for i = 1:N
    states1(i,:) = states(aircraftData.q_star, aircraftData.t(i));
    states2(i,:) = states(q_new, aircraftData.t(i));
end

% Calculate Covariance of Estimate
Rv_inv = inv(aircraftData.R_xi);

for i = 1: N
```

```
    Rv_matrix(4 * i - 3 : 4 * i, 4 * i - 3 : 4 * i) = Rv_inv;
end

covariance = inv(jacob.' * Rv_matrix * jacob);

% Pre-allocate (essentially erase)
jacob1 = zeros(4 * N, 6);
y1 = zeros(4 * N, 1);
g1 = zeros(4 * N, 1);

q_old = [470; 0.5; 0.2; 1.1; 24; 520];

% Calculate y_hat
count = 0;
flg = 0;
while(flg == 0)
    for i = 1:N
        jacob1(4 * i - 3 : 4 * i, :) = jacobian1(q_old.', aircraftData.t(i));
        y1(4 * i - 3 : 4 * i, 1) = aircraftData.Ya(:,i);
        g1(4 * i - 3 : 4 * i, 1) = states(q_old.', aircraftData.t(i));
    end

    q_new1 = q_old + inv(jacob1.' * Rv_matrix * jacob1) * jacob1.' *
Rv_matrix * (y1 - g1);

    if abs(q_new1 - q_old) < .0000000000001
        flg = 1;
    end
    q_old = q_new1;
    count = count + 1;
end

for i = 1:N
    states3(i,:) = states(q_new1, aircraftData.t(i));
end

Diagg = diag(covariance);


figure(1)
plot(states1(:,1),states1(:,2),'LineWidth', 1)
hold on
plot(aircraftData.Ya(1,:), aircraftData.Ya(2,:),'LineWidth',1,'LineStyle','--
')
plot(states3(:,1), states3(:,2),'--','color','g')
save('state_estimate','q_new1');

% Compare initial states
names = {'KNOWN INITIAL STATE', 'MEAN VALUE','VARIANCE'};
vals = table(aircraftData.q_star, q_new1, Diagg, 'VariableNames',names);
disp(vals)

% Labels
title('X & Y COORDINATES OF AIRCRAFT (SCENARIO 3)','FontSize',
25,'FontWeight','bold');
xlabel('X COORDINATES','FontSize', 25,'FontWeight','bold');
```

```
ylabel('Y COORDINATES', 'FontSize', 25,'FontWeight','bold');
legend({'TRUE PARAMETERS','MEASUREMENT DATA','ESTIMATED
TRAJECTORY'},'FontSize', 13,'FontWeight','bold','Location','NE');
grid on


function [A] = jacobian1(q,t)
% Pulls in current states and time
% Returns Jacobian at that time

q1 = q(1);
q2 = q(2);
q3 = q(3);
q4 = q(4);
q5 = q(5);
q6 = q(6);

A = [1, 0, 0, q6*(cos(q4 + (q5*t)/q6) - cos(q4)), t*cos(q4 + (q5*t)/q6),
sin(q4 + (q5*t)/q6) - sin(q4) - (q5*t*cos(q4 + (q5*t)/q6))/q6;
    0, 1, 0, q6*(sin(q4 + (q5*t)/q6) - sin(q4)), t*sin(q4 + (q5*t)/q6),
cos(q4) - cos(q4 + (q5*t)/q6) - (q5*t*sin(q4 + (q5*t)/q6))/q6;
    0, 0, 1,0,0,0;
    0, 0, 0,1,t/q6,-(q5*t)/q6^2];

end


function [res] = states(q,t)
% This function calculates your state at a specific time


q1 = q(1);
q2 = q(2);
q3 = q(3);
q4 = q(4);
q5 = q(5);
q6 = q(6);
w = q5/q6;
x_a = q6 * (sin(w * t + q4) - sin(q4)) + q1;
y_a = q6 * (cos(q4) - cos(w * t + q4)) + q2;
z_a = q3;
phsi_a = w * t + q4;

res = [x_a; y_a; z_a; phsi_a];

end
```

Problem 5

```
clc
clear
clf
close all
set(groot,'defaultfigureposition',[600 300 900 900]);
```

```
% THIS CODE WAS USED FOR PROBLEM 5

% This pulls in the q_hat estimate from problem 1 (ie run script 1 with the
% data set you want and then run this script.
q_new = load('state_estimate');

% Define Initial Variables
nameofData = "aircraftData2.mat";
load(nameofData);
dt = .1;
t = 0:dt:300;
N = length(t);
z_l = 4;
c_D = .1;

% Calculate F gains for Guidance Law
A = [zeros(3), eye(3);zeros(3), -c_D * eye(3)];
B = [zeros(3); eye(3)];
% poles = [-1.1 -1.3 -1 -1.18 -1.45 -1.2];
poles = [-.1 -.3 -.12 -.18 -.45 -.2];
F = place(A, -B, poles);

% Find Nominal State X_o for all times (plane) with q* data
% Problem 5A
plane_q_nominal = aircraftData.q_star + [0; 0; z_l; 0; 0; 0];

% Find the nominal state X_o with q_hat estimation
% Problem 5C
plane_q = q_new.q_new1 + [0; 0; z_l; 0; 0; 0];

% Preallocate
x_plane = zeros(6, N);
x_plane_c = zeros(6, N);
u_plane = zeros(3, N);
u_plane_c = zeros(3,N);
x_plane2 = zeros(4, N);

for i = 1:N
    %      5A
    %      Find Nominal Position of Plane
    x_plane(:,i) = states2(plane_q_nominal, t(i));

    %      Find Nominal Thrust Input of Plane
    u_plane(:,i) = acceleration_plane(plane_q_nominal, t(i));
    %      5C
    %      Find Position of Plane based on Guess
    x_plane_c(:,i) = states2(plane_q, t(i));

    %      Find Thrust Input of Plane
    u_plane_c(:,i) = acceleration_plane(plane_q, t(i));


    %      Nominal Plane States for video
    x_plane2(:,i) = states(plane_q_nominal, t(i));
end
```

```
% Using given q*
[x_drone, u_drone, dx] = wrapper_final(nameofData,x_plane,u_plane,F, N, dt);

% Using calculated q_hat
[x_drone_c, u_drone_c, dx_guess] =
wrapper_final(nameofData,x_plane_c,u_plane_c,F, N, dt);

% PART A PLOT
figure(1)
sgtitle('Problem 5A (SCENARIO 1)','FontSize', 30,'FontWeight','bold');
% POSITION
subplot(2,1,1)
plot(t,dx(1,:),'LineWidth',2)
hold on
plot(t,dx(2,:),'LineWidth',2)
plot(t,dx(3,:),'LineWidth',2)

title('ERROR IN DRONE POSITION','FontSize', 25,'FontWeight','bold');
xlabel('TIME (s)','FontSize', 25,'FontWeight','bold');
ylabel('POSSITION ERROR', 'FontSize', 25,'FontWeight','bold');
legend({'\DeltaX','\DeltaY','\DeltaZ'},'FontSize',
13,'FontWeight','bold','Location','NE');
% VELOCITY
subplot(2,1,2)
plot(t,dx(4,:),'LineWidth',2)
hold on
plot(t,dx(5,:),'LineWidth',2)
plot(t,dx(6,:),'LineWidth',2)

title('ERROR IN DRONE VELOCITY','FontSize', 25,'FontWeight','bold');
xlabel('TIME (s)','FontSize', 25,'FontWeight','bold');
ylabel('VELOCITY ERROR', 'FontSize', 25,'FontWeight','bold');
legend({'\DeltaV_x','\DeltaV_y','\DeltaV_z'},'FontSize',
13,'FontWeight','bold','Location','NE');

figure(2)
sgtitle('Problem 5C (SCENARIO 1)','FontSize', 30,'FontWeight','bold');
% POSITION
subplot(2,1,1)
plot(t,dx_c(1,:),'LineWidth',1)
hold on
plot(t,dx_c(2,:),'LineWidth',1)
plot(t,dx_c(3,:),'LineWidth',1)

title('ERROR IN DRONE POSITION','FontSize', 25,'FontWeight','bold');
xlabel('TIME (s)','FontSize', 25,'FontWeight','bold');
ylabel('POSSITION ERROR', 'FontSize', 25,'FontWeight','bold');
legend({'\DeltaX','\DeltaY','\DeltaZ'},'FontSize',
13,'FontWeight','bold','Location','NE');
% VELOCITY
subplot(2,1,2)
plot(t,dx_c(4,:),'LineWidth',1)
hold on
plot(t,dx_c(5,:),'LineWidth',1)
plot(t,dx_c(6,:),'LineWidth',1)

title('ERROR IN DRONE VELOCITY','FontSize', 25,'FontWeight','bold');
```

```
xlabel('TIME (s)','FontSize', 25,'FontWeight','bold');
ylabel('VELOCITY ERROR', 'FontSize', 25,'FontWeight','bold');
legend({'\DeltaV_x','\DeltaV_y','\DeltaV_z'},'FontSize',
13,'FontWeight','bold','Location','NE');

function [x_drone, u_drone, dx] = wrapper_final(nameofData,x_plane,u_plane,F,
N, dt)

% THIS CODE WAS USED FOR PROBLEM 5

load(nameofData);

% Preallocate

x_drone = zeros(6, N);
dx = zeros(6, N);
du = zeros(3, N);
u_drone = zeros(3,N);


% Find X for all times (drone)
x_drone(:,1) = [13 12 112 0 0 0].';
u_drone(:,1) = [.4 .6 .5].';
dx(:,1) = x_drone(:,1) - x_plane(:,1);
p_drone = [ [.9 0 0; 0 .8 0; 0 0 .5], zeros(3); zeros(3) .2 * eye(3)];



for i = 2:N
    %     calculate position/covariance of drone
    [x_drone(:,i), p_drone] = kalman(x_drone(:,i-1), u_drone(:,i-1), p_drone,
dt);

    % Find dx (drone - plane)
    dx(:,i) = x_drone(:,i) - x_plane(:,i);

    % Find du based on F * dx
    du(:,i) = F * dx(:,i);

    %     update u_drone
    u_drone(:,i) = du(:,i) + u_plane(:,i);
end


end


function [x_k, P_k] = kalman(x_prior, u_prior, P_prior, T)

% THIS CODE WAS USED FOR PROBLEM 5

% Given Value
C_d = .1;
```

```
% Prior Estimates
phi_k = [eye(3), T * eye(3); zeros(3),eye(3)*(1-C_d*T)];
B = [zeros(3); T*eye(3)];
C = [eye(3), zeros(3)];
Rv = [1 0 0; 0 1 0;0 0 .8];

% Estimate Y_k
x = droneDynamics(x_prior,u_prior,T);
y_k = C * x + [normrnd(0,1,1);normrnd(0,1,1);normrnd(0, .8,1)];

% Prediction
x_guess = phi_k * x_prior + B * u_prior;
p_guess = phi_k * P_prior * phi_k.';

% Kalman Gain
K_k = p_guess * C.'* inv(C * p_guess * C.' + Rv);

% Update estiamte with y_k measurement
x_k = x_guess + K_k * (y_k - C * x_guess);

% Update Error Covariance
P_k = (eye(6) - K_k * C) * p_guess;

end

function [res] = states2(q,t)
% This function calculates your state at a specific time
q1 = q(1);
q2 = q(2);
q3 = q(3);
q4 = q(4);
q5 = q(5);
q6 = q(6);
w = q5/q6;

% Calculate Position Based on Integrated Velocity
x_a = q6 * (sin(w * t + q4) - sin(q4)) + q1;
y_a = q6 * (cos(q4) - cos(w * t + q4)) + q2;
z_a = q3;

% Calculate Velocity Based on Closed Form Equations Given
Vx = q5 * cos(w * t + q4);
Vy = q5 * sin(w * t + q4);
Vz = 0;


res = [x_a; y_a; z_a; Vx; Vy; Vz];

end
```