

Module 1 Assignment

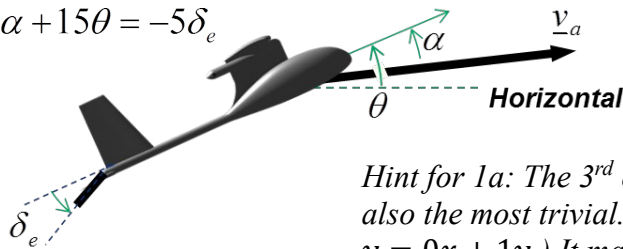
See final page for homework submission expectations.

525.661 Unmanned Aerial Vehicles Systems and Control

1) In an airplane, the motion of angle-of-attack (α), pitch rate ($\dot{\theta}$) and pitch (θ) are tightly related and respond very quickly to a change in the “elevator” control surface (δ_e). (In later classes we will derive and utilize these relationships). Consider a dynamic system satisfying:

$$\dot{\alpha} + 3\alpha - 2\dot{\theta} = 0$$

$$\ddot{\theta} + 10\alpha + 15\theta = -5\delta_e$$



Hint for 1a: The 3rd equation may be the trickiest, but is also the most trivial. (Consider, for example, $y = 0x + 1y$.) It may seem trivial, but is necessary for (b)

a) Re-arrange the above two equations to form three equations with the following form:

$$\dot{\alpha} = ?\alpha + ?\dot{\theta} + ?\theta + ?\delta_e$$

$$\ddot{\theta} = ?\alpha + ?\dot{\theta} + ?\theta + ?\delta_e$$

$$\dot{\theta} = ?\alpha + ?\dot{\theta} + ?\theta + ?\delta_e$$

b) Let the state vector be: $\underline{x} = [\alpha \quad \dot{\theta} \quad \theta]^T$

Convert the above dynamic model into a SISO state space model going from an input of δ_e to an output of θ . (i.e. determine the A, B, C, and D matrices)

c) Using Matlab, convert the resulting state space model into a transfer function in Zero-Pole-Gain form.

For checking:

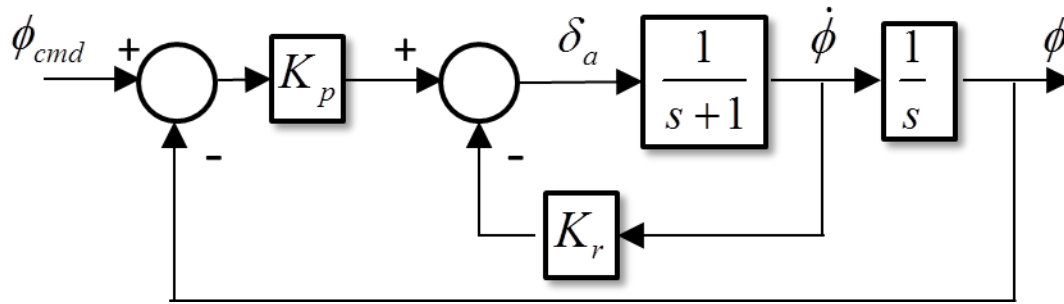
$$\frac{-5 (s+3)}{(s+1.373) (s^2 + \text{XXXX}s + 32.77)}$$

d) Modify the Matlab script example provided in the lecture slide “State Space response example using Matlab” to generate a time-response of the state-space model from part (b) to a unit-step elevator input (i.e., $u = \delta_e = 1$). Start the script with an initial state vector of $\underline{x} = [0 \ 0 \ 0]^T$, and propagate time from 0 to 5 seconds. Turn in the script and compare the result with a step response of the transfer function you generated in part (c):

```
plot(tHistory,yHistory, tHistory,step(G,tHistory), 'r--');
```



2) Consider the following block diagram that represents a roll control autopilot. In the diagram, the $1/(s+1)$ block represents the “plant” transfer function from the aileron deflection (δ_a) to roll rate ($\dot{\phi}$). As we’ll see, a roll control autopilot might be designed using a roll rate feedback gain (K_r) and a proportional gain on the roll error (K_p).



- Determine the closed-loop transfer function for the roll control autopilot shown. (Hint: it should look like a 2nd order model)
- For the autopilot to control roll in a stable manner, an autopilot designer must appropriately choose the values K_r and K_p . Determine the gains K_r and K_p to achieve a desired response of: $\omega_n=2.5$ and $\zeta=0.6$

For checking:
 $K_r * K_p = 12.5$

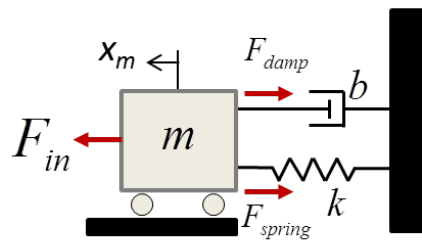
- Using “`s=tf('s')`” in Matlab, make the closed loop transfer function using your chosen gains.
- Use the “step” function to plot the resulting step response from roll command to achieved roll.

- 3) Consider a spring-damper system with the following values:

$$m = 2 \text{ kg}$$

$$k = 12 \frac{\text{N}}{\text{m}}$$

$$b = 1 \frac{\text{N}}{\text{m/s}}$$



x_m : Mass Pos.

m : mass

$$F_{damp} = b\dot{x}_m$$

$$F_{spring} = kx_m$$

F_{in} = Input Force

- Determine the transfer function (from input force to mass position), and use it to determine the natural frequency and damping. What should be the steady-state position given a unit step force input?
- Using `s=tf('s')` and the `step()` function, plot the input-output step response from input force to mass position. Make sure your step response characteristics match your answers to (a).
- What would be the steady-state mass position, given a constant input force of 10 N (for $t \geq 0$).
- Consider a changeable damper, meaning you have the freedom to change the value of “b”. What value of “b” would result in a damping of 0.7? What value of “b” would result in a critically damped system? Compare both step responses with the step response from the original $b=1 \text{ N/(m/s)}$ system.
- Using the symbols m , b & k , model the 2-state spring damper system as a state-space model with the states of position and velocity (i.e. determine the matrices A , B , C & D as functions of the symbols m , b & k).

$$\begin{bmatrix} \dot{x}_m \\ \ddot{x}_m \end{bmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} x_m \\ \dot{x}_m \end{bmatrix} + \begin{bmatrix} ? \\ ? \end{bmatrix} F_{in}$$

$$\text{mass position} = \begin{bmatrix} ? & ? \end{bmatrix} \begin{bmatrix} x_m \\ \dot{x}_m \end{bmatrix} + \begin{bmatrix} ? \end{bmatrix} F_{in}$$

4) Throughout the semester we'll be developing a UAV 6DOF in Simulink/Matlab. The Simulink architecture of the 6DOF will be provided to you for download from the class site, along with "stubbed" Matlab functions. To get started with our simulation, we need some conversion utilities. Using the function stubs provided in the downloadable .zip file, create the following matlab functions:

Hint: Use 3 separate matrices:

$$R_{NED}^b = R_x(\phi)R_y(\theta)R_z(\psi)$$

`R_ned2b = eulerToRotationMatix(phi_rad, theta_rad, psi_rad)`
Make a 3x3 rotation matrix from ned to body coordinates

Assume order: 1) psi_rad about z, 2) theta_rad about y, 3) phi_rad about x

`[phi_rad theta_rad psi_rad] = rotationMatrixToEuler(R_ned2b)`

Assume same order as above

Handle theta_rad = $\pm\pi/2$ case ($\pm 90^\circ$)

`[Va alpha_rad beta_rad] = makeVaAlphaBeta(v_rel_b)`

$\underline{v}_{rel_b} = \underline{v}_a^b$: wind-relative airspeed vector in body coords

Handle Va=0 case (i.e. set alpha_rad and beta_rad to zero)

`[Vg gamma_rad course_rad] = makeVgGammaCourse(v_ned)`

\underline{v}_{ned} : groundspeed vector in NED coords

Handle Vg=0 case (i.e. set gamma_rad and course_rad to zero)

Remember to use
`atan2(num,den)` !!!

You don't need to turn in the code. See below for test case results to submit.

Test cases to submit:

a) Compute `R_ned2b` for:

phi_rad = $14 \cdot \pi / 180$; (14 degrees)

theta_rad = $-30 \cdot \pi / 180$; (-30 degrees)

psi_rad = $160 \cdot \pi / 180$; (160 degrees)

b) Compute Euler angles from the `R_ned2b` computed in (a). (The function computes angles in radians, but provide your answers in degrees. If you didn't get the original values back, you made a mistake.)

c) Extract V_a , α and β from $\underline{v}_a^b = [20 \ -2 \ 3]^T$. Routine generates angles in radians, but provide your answers in degrees. (For checking: $\beta = -5.6478^\circ$)

d) Extract V_g , γ and course from $\underline{v}_g^{ned} = [-15 \ -12 \ 2]^T$. Routine generates angles in radians, but provide your answers in degrees. If this is the velocity vector for a UAV, is it climbing or descending? Approximately what direction is the vehicle heading (N, NE, E, SE, S, SW, W, NW)?

e) A UAV is flying with an attitude of $\phi=12^\circ$, $\theta=2^\circ$ and $\psi=-140^\circ$, and a ground-relative velocity vector of $\underline{v}_g^{ned} = [-15 \ -12 \ 2]^T$ m/s. The local wind vector is $\underline{v}_w^{ned} = [3 \ 4 \ 0]^T$ m/s. Generate \underline{v}_a^b . Then, generate V_a , α and β . (Express angles in degrees). (For checking: $\beta = 2.9929^\circ$)



5) Now that you have some necessary working routines, you should be able to run the Simulink simulation, `uavsim.mdl`. To verify, do the following:

- a) The simulation is loaded via the routine “`load_uavsim`”. Run `load_uavsim` and verify that the sim opened and was prepared without error. (The Simulink model should open.) (Yes/No)
- b) The workspace should contain a structure “P” which contains all the parameters necessary to run the simulation. In addition to opening the Simulink model, “`load_uavsim`” calls “`init_uavsim_params`”, which creates the “P” structure. Likely, none of these parameters make sense to you yet, but they will! Explore “`init_uavsim_params`” and answer the following questions, including units:
 - What is the vehicle mass?
 - What is the maximum allowable roll command? (in degrees)
 - What is the nominal airspeed?
- c) Run the Simulink model by pressing the “play” button (▶). The number next to the “play” button shows how long the simulation will run. You should see a UAV displayed, but it won’t be moving and it will have zero roll and zero pitch. But, the displayed time on the figure should be updating. Did it run without errors? What is the vehicle altitude in meters?
- d) The “P” structure contains parameters and initializations, including the initial attitude angles, `P.phi0`, `P.theta0`, and `P.psi0` (all in radians). If you coded the Euler/RotationMatrix functions correctly, you should be able to change these values and see a difference in the visualization. In the Matlab command console, type “`P.phi0 = 90*pi/180;`”, then re-play the Simulink model. Did the visualization change appropriately?

6) For each of the angles below, state:

- Variable used to represent quantity (Greek letter and the English spelling of the Greek letter. E.g. χ and chi, or α and alpha)
- Description of positive direction (e.g. +EastOfNorth)

Turn in a table similar to:

<i>Angle</i>	<i>Greek Symbol (e.g. α)</i>	<i>English Spelling (e.g. alpha)</i>	<i>Description of Positive Direction</i>
Roll			
Pitch			+: Nose Up
Yaw			
Angle-of-Attack			
Sideslip			
Course			
Vert. flight path angle			
Crab angle			



Homework Submissions

We have high expectations for submitted work in this course because engineers working toward a Master's Degree should be able to regularly demonstrate high quality work and communicate their work effectively.

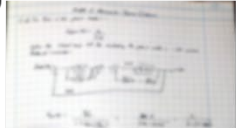
Assignment submission guidelines for all homeworks in 525.661:

- Entire submission should be contained in a single organized pdf.
 - .docx is acceptable, but .pdf is preferred
 - You may optionally choose to provide additional mfiles, etc., with your submission, but we aren't grading those and won't be looking at them.
- Submission should be organized by problem number, and all requested information in a particular problem (*math, responses, analyses, answers, figures, Matlab code, etc.*) should be together in the pdf and in a logical organized manner.
 - To be specific, if problem 18(d) requests some hand-written math, Matlab code and a figure, then all of those artifacts should be in the "18(d)" portion of your submission. i.e. we shouldn't have to scroll to another part of the pdf to search for the hand-written work, code, etc.
- Requested information can be hand-written or typed, and math can be worked out by hand or using Matlab or other tools. Either way, the burden is on the student to sufficiently show that they acquired the result through an appropriate procedure.
- Provided figures should be labeled appropriately and specifically (using xlabel, ylabel, title, and legend where applicable). Figures should always be scaled or zoomed appropriately to naturally show the "dynamics" that we will obviously be looking for.
 - e.g. if a step response is vertically zoomed out so that we can't see the response details we are looking for, then it won't receive credit
- When we ask for figures/plots/time-histories/step-responses/etc. to be compared, the burden is on the student to plot them in a way that can be compared appropriately while still seeing the detail necessary. For some comparisons, this might mean overlaying plots in the same figure, provided that all the expected wiggles are still visible. For other comparisons, it might mean aligning appropriately scaled figures vertically or horizontally so that the desired comparisons or differences are evident. *Use your engineering judgement!*
- Many, many problems have built-in "checks" (e.g. Problem 1(c) in HW1.) Use them. Also use your engineering sense: If an instructor asks a student to compare two things or to do something two different ways, then it is probably because the results are either supposed to compare well, or are supposed to highlight some difference. Use such insight to self-check whether you are on the right track.
- Review your submission in the final pdf format before turning it in.
 - Always verify that you are providing all the information that we ask for.
 - We do not have Canvas set up to allow multiple submissions.
- There is a 10% per day late penalty, unless arrangements were made with instructors before-hand. But you do the math! A 1-day penalty might be better than an incomplete or erroneous on-time submission.

Example:

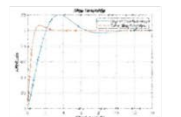
Problem 1

1a)



1b)

```
clear; %clear workspace
%Define system
%Transfer function
num = 1;
den = [1 1];
%Discrete transfer function
[numz,denz] =impinvar(num,den,10);
%Step response
t = 0:0.1:10;
[~,h] = stepz(numz,denz,t);
plot(t,h,'b');
hold on;
%Plotting
xlabel('Time (s)');
ylabel('Response');
title('Step Response');
legend('Step Response');
```




1c)

Let $x(t)$ be the response of the system to a unit step input. Plot $x(t)$ for $t \in [0, 10]$ and compare it with the response of the system to a unit impulse input. Use the `impz` function to compute the impulse response.

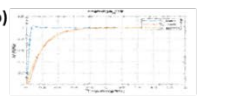
Let $y(t)$ be the response of the system to a unit step input. Plot $y(t)$ for $t \in [0, 10]$ and compare it with the response of the system to a unit impulse input. Use the `impz` function to compute the impulse response.

Problem 2


2a)



2b)



Problem 3



Problem 4

4a)

```
clear; %clear workspace
%Define system
%Transfer function
num = 1;
den = [1 1];
%Discrete transfer function
[numz,denz] =impinvar(num,den,10);
%Step response
t = 0:0.1:10;
[~,h] = stepz(numz,denz,t);
plot(t,h,'b');
hold on;
%Plotting
xlabel('Time (s)');
ylabel('Response');
title('Step Response');
legend('Step Response');
```

4b)

