

MATLAB Group B Project Report

EEL3135 - Signals and Systems

Spring 2021

Instructor: O. Patrick Kreidl

Lab Report Due: April 23rd, 2021

Nicholas Myers - N01431456

Christopher Merring - N01122482

Jandi Clark - N01425456

Caleb Suit - N01422143

**Abstract:**

**Exercise 1 (Transfer Function Analysis):** Exercise 1 explored MATLAB’s numerous commands for navigating transfer functions, encompassing tools necessary for handling polynomials, ratios of polynomials, and partial fraction expansions as well. In Exercise 1, MATLAB was used to generate a transfer function and display the results in 2 formats, and also to calculate a partial fraction expansion in multiple forms for both a continuous-time transfer function and a discrete-time transfer function.

**Exercise 3 (Automation and Control):** Exercise 3 explored the use of feedback sensors in regards to altitude measurements for regulating vertical thrust on an aerial vehicle. A thrust waveform was designed and MATLAB was utilized to calculate the veracity of the waveform in regards to altitude control. The exercise explored various iterations of vehicle mass, and also compared the use of a closed-loop system to an open-loop system.

**Results and Observations**

**Exercise 1:**

This exercise involved using MATLAB’s system simulation capabilities to analyze continuous and discrete-time system transfer functions.

*Part A*:

A continuous-time transfer function *H(s),* expressed as a ratio of factors of s, is shown in *Fig. 1*. In order to format the expression as a ratio of two polynomials, MATLAB’s numerical computing abilities can be used to avoid lengthy hand calculations. MATLAB’s expression for the transfer function *H(s)* in the ratio of factors form is shown in *Fig. 2* and *Fig. 3*. Two different formats are provided for the sake of readability. Converting this expression into a ratio of two polynomials can be done by using the built-in *expand()* function; this produces the rearranged transfer function *H(s)* given in *Fig. 4* and *Fig. 5*.

|  |
| --- |
|  |
| *Fig. 1:*A continuous-time transfer function *H(s),* expressed as a ratio of factors of s |
|  |
| *Fig. 2:* MATLAB format 1 of transfer function *H(s).* |
|  |
| *Fig. 3:*MATLAB format 2 of transfer function *H(s).* |
|  |
| *Fig 4:* Transfer function *H(s)* presented as a ratio of two polynomials in MATLAB format 1. |
|  |
| *Fig 5:* Transfer function *H(s)* presented as a ratio of two polynomials in MATLAB format 2. |

*Part B:*

When attempting to find the impulse response of a transfer function, it is often necessary to expand an expression composed of polynomials into a summation of fractions which lend themselves easily to table lookups. Such is the case with the transfer function given in Part A. MATLAB’s *partfrac()* function was applied to the transfer function *H(s)* and verified using a plot. The partial fraction expansion of *H(s)* is shown in *Fig. 6* and *Fig. 7*, and the plot is provided in *Fig. 8.*

|  |
| --- |
|  |
| *Fig. 6:* Partial fraction expansion of *H(s)* in MATLAB format 1. |
|  |
| *Fig. 7:* Partial fraction expansion of *H(s)* in MATLAB format 2. |
|  |
| *Fig 8:* Plot verifying the partial fraction expansion of *H(s).* |

*Part C:*

MATLAB’s system simulation abilities extend to discrete-time systems as well; these capabilities are explored through analysis of the DT transfer function *H[z]* shown in *Fig. 9.H[z]* is provided in the form of a ratio of two polynomials, which is displayed in MATLAB’s output in *Fig. 10* and *Fig. 11.* To put this expression in the form of a ratio of factors, the *roots()* function can be called upon; *roots()* provides the roots of a given polynomial expression, and is helpful in this conversion. The rearranged transfer function *H[z]* is provided in *Fig. 12* and *Fig. 13*.

Following the steps of Part B, the partial fraction expansion of *H[z]* can be determined using built-in systems tools. The *residuez()* function returns coefficients of the partial fraction expansion of a DT system input. The function is particularly useful in this exercise as it returns the expression’s residues and poles as a matrix, allowing for for-loop integration to quickly format the resulting expression. The partial fraction expansion of the transfer function is shown in terms of *z* in *Fig. 14* and *Fig. 15*, and in terms of in *Fig. 16* and *Fig. 17.*

When considering the *impulse()* command as applied to these systems, if it is already known that the systems are stable, then the command will simply verify that an input of finite energy produces an output of finite energy, thus rendering the *impulse()* command useless for verifying this fact.

|  |  |
| --- | --- |
|  | |
| *Fig. 9:* Discrete-time transfer function *H[z]*. | |
|  |  |
| *Fig. 10:* MATLAB format 1 of transfer function *H[z].* | *Fig. 11:* MATLAB format 2 of transfer function *H[z]*. |
|  |  |
| *Fig. 12:* MATLAB format 1 of transfer function *H[z]* after use of roots(). | *Fig. 13:* MATLAB format 2 of transfer function *H[z]* after use of roots(). |
|  | |
| *Fig. 14*: MATLAB format 1 of partial fraction expansion of *H[z]* in terms of z. | |
|  | |
| *Fig. 15*: MATLAB format 2 of partial fraction expansion of *H[z]* in terms of z. | |
|  | |
| *Fig. 16*: MATLAB format 1 of partial fraction expansion of *H[z]* in terms of . | |
|  | |
| *Fig. 17*: MATLAB format 2 of partial fraction expansion of *H[z]* in terms of . | |
|  | |
| *Fig. 18:* Plots verifying partial fraction expansion of *H[z]*. | |

**Exercise 3:**

This exercise involved the analysis of a system regulating the altitude of an aerial vehicle and the design of feedback control to stabilize the system. With the altitude of the vehicle denoted *y(t)*, the vertical thrust denoted *x(t)*, and the vehicle’s mass denoted *M*, the following LTI model of the system was given as *P(s) = Y(s)/X(s) = 1/(Ms2).*

*Part A:*

Assuming the vehicle mass *M* to be 1 gram, a thrust waveform was designed to smoothly raise the vehicle’s altitude to 2 meters in one second, and then to have the vehicle hover at that altitude indefinitely. Given that the system acts as a double integrator, the thrust waveform was calculated by taking the second derivative of a derived expression for appropriate altitude over time. The target output waveform was chosen to be

.

This expression is a summation of a positive quadratic between 0 seconds and 0.5 seconds, a negative quadratic between 0.5 seconds and 1 second, and a constant after 1 second. The function produces the waveform shown in *Fig. 29.* To get the thrust waveform *x(t)*, the target waveform equation is reduced to its second derivative:

.

|  |
| --- |
|  |
| *Fig 29.* Target output waveform for thrust assuming vehicle mass of 1 gram. |

*Part B:*

To show that the derived thrust waveform from Part A produces the expected altitude waveform, MATLAB was used to simulate the system with mass *M* = 1 and *M* = 0.5. The thrust waveform was simulated alongside the two transfer functions for the system, and the results are shown in *Fig 30.*

The simulation produces a viable result for *M =* 1, with the altitude climbing smoothly to 2 meters over a single second, then leveling off at 2 meters for some time afterwards. In the case of *M* = 0.5, the altitude climbs to 4 meters over a single second, then levels off at that height. Because the input waveform is dependent on vehicle mass, a vehicle of unknown mass may behave unpredictably when the given thrust waveform is applied.

|  |
| --- |
|  |
| *Fig. 30:* Results of thrust waveform of transfer function simulation where M = 1 is shown in blue and M = 0.5 is shown in orange. |

*Part C:*

For Part C, a feedback control loop was introduced into the system. The constant *K* denotes a tunable signal gain, and *F(s)* represents a proportional-derivative control unit with the transfer function *F(s) =* 1 *+ αs*. A graphical representation of the feedback loop taken from the EEL3135 MATLAB project document is shown in *Fig. 31.* Using *M* = 1, the constants *α* and *K* were chosen such that the transfer function’s poles lie at -5 ± j√5. The derivations for calculating *α =* 10 and *K =* 30 are provided in *Fig. 32*. Next, the amplitude *A* of a step command *r(t) = Au(t)* was derived using MATLAB’s ilaplace() function such that the steady state value of the output is equal to 2 meters in altitude. The derivation of this is shown in *Fig. 33.*

|  |
| --- |
|  |
| *Fig. 31:* Graphical representation of feedback control loop. |
|  |
| *Fig. 32:* Handwork showing calculation of *α =* 10 and *K =* 30. |
|  |
| *Fig. 33:* Handwork showing derivation of Amplitude A of step command *r(t) = Au(t)*. |

*Part D:*

To verify that the step command seen in Part C produces a viable output, MATLAB’s simulation tools were used to simulate the system with both *M =* 1 and *M =* 0.5; the output waveforms simulated is shown in *Fig 34.*

|  |
| --- |
|  |
| *Fig. 34:* System simulation verifying Exercise 3.c step command. Blue line represents *M* = 1 and the orange line represents *M =* 0.5: |

*Part E:*

To compare the thrust waveform inputs to the double integrator system P(s) for the open and closed loop systems, the transfer function between the Laplace transform of the step command r(t) and the thrust waveform x’(t) was derived. This produced the following equation for the thrust waveform: (400\*exp(-5\*t)\*(cos((5^(1/2)\*t)/2) + (19\*5^(1/2)\*sin((5^(1/2)\*t)/2))/20))/21 - (16\*exp(-5\*t)\*((5\*cos((5^(1/2)\*t)/2))/4 + (19\*5^(1/2)\*sin((5^(1/2)\*t)/2))/16))/21 - (160\*exp(-5\*t)\*((19\*cos((5^(1/2)\*t)/2))/8 - (5^(1/2)\*sin((5^(1/2)\*t)/2))/2))/21. The MATLAB equation is shown in *Fig*. *35.*

The two thrust input waveforms were graphically compared, and the plot is given in *Fig. 36.*

|  |
| --- |
|  |
| *Fig. 35:* MATLAB equation showing transfer function between laplace transform of r(t) and thrust waveform x’(t). |
|  |
| *Fig. 36:* Open versus closed loop input plot. Part A input is red orange for M=0.5 and blue for M=1, and Part E input is yellow for M=0.5, and purple for M=1. |

*Part F:*

As can be seen in the above analysis, using a closed loop setup for controlling the altitude of the vehicle offers several advantages over the open loop system seen in Part A. The most obvious advantage that the closed loop system provides is making the altitude output much less sensitive to changes in the vehicle mass. When the mass of the vehicle was reduced by 50% in Part A, the altitude output leveled off at twice the target altitude, while the same mass reduction performed in Part D resulted in no altitude difference, stabilizing at 2 meters over a single second.

Another advantage that the feedback system offers is a smoother transition from the starting to the target altitude. The thrust waveform designed in Part A features a summation of functions weighted with the unit step function, resulting in discrepancies between the transition points of the summed functions; the feedback system, however, provided a thrust waveform composed of a single function independent of unit step or unit impulse weights. This results in a continuous waveform which smoothly raises the vehicle’s altitude at reasonable accelerations.

## 