**Controls Analysis**

**of Tepid Water Heater**

**By**

**Alex Snyder**

**And**

**Patrick Shurina**

**Executive Summary**

The purpose of this report is to analyse a control system included in a modern tepid water heater. For this purpose we chose an electrically heated tank type water heater. These types of water heaters are widely used in public restrooms, and in safety apparatuses.

The water level, temperature, pressure, and outward flow rate are all controlled in this system. The level by a dipstick or bober controller similar to that in a toilet tank. The temperature by thermostats and heating elements. The pressure by a control valve, and in some older tanks a rupture disk. Flowrate and outgoing pressure are controlled by PCV and FCV systems.

The transfer function used in this paper is defined by several variables, and is very complex. In the analysis portion we will focus on a small part of the transfer function known as P(s) that focuses on the capacitance, heater constant, and thermal resistance.

Various tests were carried out to observe changes to peak amplitude, peak time, rise time, and setting time. The tests included varying capacitance, thermal resistance, and heater constant. Tests were also conducted to see the effect of adding an extra pole or zero to the system would have on the considered variables.

MatLab and Simulink software was used to carry out the tests and create graphical representations of the data collected. Under all conditions tested the function remained stable. Changing k changes amplitude. If R is changed, all variables of interest are changed. If C is increased or decreased it has a positive correlation with rise, peak and setting time. If negative poles or zeros are added they affect peak amplitude without affecting rise, peak, and setting time.

Further study is recommended in regard to the feasibility of adding poles and zeros, examining other parts of the equation, and examining the overall equation rather than individual parts.

**Table of Contents page #**

1. **Introduction 1**
2. **Control System 1-3**
3. **Test Plan and Model 4-5**
4. **Results 5-17**
5. **Conclusions 17**
6. **Recommendations 17**
7. **References 18**
8. **Appendix A 19**
9. **Appendix B 20-45**

**Introduction**

The Tepid water heater is a critical heat exchanger device that is applied in many types of industries. The term tepid implies that the water is only slightly heated staying between the saturated water and saturated vapor lines of a T-V diagram. This system is important because tepid water heaters are used in motion sensored sinks, emergency showers, and eyewash stations. Tepid water heaters can also be beneficial financially because the heat transfer that takes place is less because the desired change temperature is less. This implies the cost of heating the water in a heat exchanger type of system is also less. There are four main kinds of tepid water heaters; tank, tankless, hybrid, and point to point. The decision was made to focus on the tank type water heater because it is the most common of the four. Secondly, tank type water heaters also can be separated between gas and electrically heated. Due to the efficiency of the electrically heated water heater and its everyday use in most homes and facilities it was chosen. Figure one that is illustrated below depicts the complex control system that was used as the control system for the tepid water heater.

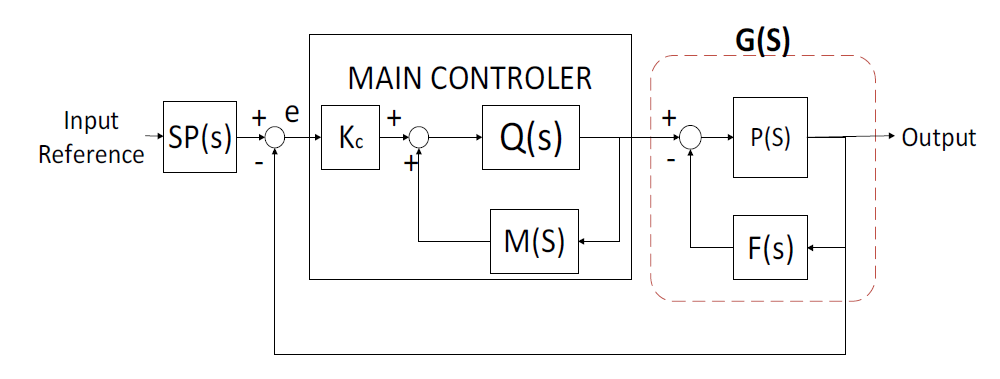


Figure 1: Tepid water heater block flow diagram

**Control System**

**How the control system works:**

The schematic for the water heater can be seen below in Appendix A. The system starts by calling for domestic water. The system knows to ask for domestic water because of the dipstick and limit switch devices. The dipstick ,just like the one that measures oil in a car, is set to limit switches that read the level of the water in the tank. When the water becomes too low the system turns on and asks for domestic water to fill to a certain limit. If the water trips the upper limit the drain valve, if level controlled, will turn from close to open to drain necessary water. If the tank does not have a LCV an overflow pipe is used to drain excess water. A manual drain valve is also attached at the bottom of the tank so that the tank can be maintained and moved with ease. Once the water is inside the tank the lower and upper elements are used to heat and maintain the water to a specific temperature. This system acts as a heat exchanger where heat is transferred from the heating element to the water. Thermostats that change the temperature can be controlled on the face of the tank and or through a computer system. When water is called for from a secondary location water is then sent through hot water piping where the flow rate and pressure can be measured using either FCV and PCV or in a simpler system a pressure gauge. FCV and PCV have the ability to partially open and close valves to increase or decrease the rate and pressure at which the water is flowing through them.Flow rate is important because the velocity is a large component when considering heat transfer, if the rate is slow more heat is able to transfer from the heating element to the water. In a simpler system pressure gauges are generally controlled by a manual butterfly type isolation valve that would need to be hand operated to increase or decrease the pressure or flow. Also standard on tepid water heater systems is a pressure release valve, this valve monitors the pressure inside the tank. When these tanks are produced there is a set limit which is within safe operating range. If the tank becomes over pressurized it will open to relieve the build up pressure and close when the pressure has returned within the operating limits. These three control systems work in conjunction with one another to perform the task of sending water to a secondary location.

**Derivation of the transfer function:**

Equations 1 through 11 as well as figure 1 are provided by the paper by Hondianto et al. while the following transfer function equation is derived from figure 1 and equations 1 through 11 [1]. These equations are needed to understand the transfer function below, and can be substituted into the transfer function as needed.

From Figure 1 and the following equations it is possible to find the transfer function. The variables used are K, which is the gain associated with the controller, T defined as the time constant, L which is the dead time, as well as k which is the heater constant, C, the thermal capacitance, and R, the thermal resistance, λ is related to the speed of response, and ɑ relates to disturbances [2].



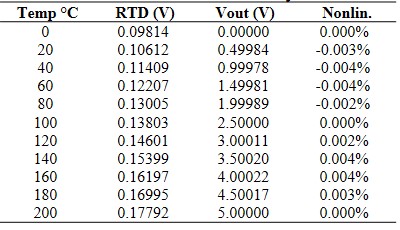
The transfer function of the system which has been previously generated for figure # is shown below.

****

**Focusing on the P(s) Function:**

Due to the complexity of the transfer function, this report will specifically focus on the P(s) function which is the initial and simplest version of the tepid water heater transfer function that was provided by the paper by Hondianto et al. This simplification illustrates how electrical elements such as capacitance, heater constants, and resistances affect the stability of the tepid water heater. From these components, similarly to a thermocouple, a voltage output can be obtained and related to a temperature as shown in table 1.

Table 1: shows output voltage related to temperature

****

**Test Plan and Model**

**Transfer Function:**

As stated above the P(s) transfer function was focused on for the stability analysis. This allowed for a more in depth analysis and more variable alteration in seeing how the stability of the system could be changed.

**Tests That Occured:**

One test that occured was changing the capacitance value and observing how it changed the peak time , rise time, and settling times of the system if all other variables are held constant. This test also observed the output of the system as a result of changing an input to the system. This test was able to show the limits of the system, and how larger or smaller input affects the shape, size, and ossicalations of the frequency vs time graph. This is important because hot water from a tepid water heater needs to be provided as quickly and consistently as possible. A desired rise time , settling time, and peak time can be used as limits and then a range of acceptable input capacitance was determined.

Secondly, The value of R which takes into account thermal resistance was altered so that the response to the system was observed while other input values were held constant at different R values. This showed how the thermal resistance affected the time taken to reach desired output values, which in turn affected rise time, peak time , and the settling time. Realistic values were chosen to provide possible alterations to the current tepid water heater control systems to increase stability. This test is important because different components can be used to provide different levels of thermal resistance, from this test it was determined which values of R are needed to provide the quickest and most stable transition from input temperature to output temperature.

The third test conducted was to change the value of k, which is the heater constant, and show how it changed the systems output. Similarly to the changing the R value in the second test, it showed how different realistic strengths of heating elements affected the peak time , rise time, and setting time. The alteration of k determined which k values provide the fastest and most stable response to the system.

The fourth test was adding a pole at different values and observing how it changes the response and output of the system as all other variables are held constant. The effects from adding a pole was observed by determining the change in rise time, setting time, and output temperature as the peak time is changed. This test showed how the addition of poles changed the response of the system andl affected the stability of the system. It determined if a larger or smaller pole created a larger amount of oscillations and a longer distance to reach settling time which demonstrates the instability of the system. This allowed for observations on the overshoot that occured and determined if the overshoot amount increased or decreased as the size of the pole of the system increased or decreased.

Similarly to the fourth test, a fifth test was conducted to observe how adding a zero affected the overall output of the system including rise time, settling time, peak time. These values were used to determine the stability of the system.

Each test used 10 separate values, 5 increments above the initial value and at least 4 increments below the initial value if the initial value is omitted than 5 values below the initial value were used. Initial values were determined from the Hondianto et al. text. Values are listed in the table below.

Table 2: Original values of transfer function

| K-heater constant | C-Capacitance | R-Resistance |
| --- | --- | --- |
| .0464 | 1 Farad | 77.58 ohms |

**Administration of testing:**

Matlab and Simulink software were used to complete testing and display graphical results, specifically the linear system analyzer, TF(num,den), and PZmap commands were used to test the stability of the P(s) transfer function alterations. The value of P(s) is shown above in equation 1. The full matlab code and complete results are shown in appendix section B.

**Results**

The first test that was carried out was to vary the heater constant and observe how the system changed. The original value was given above, from table 2, as 0.0464 and varied by 0.01 for each step. The original value plus the five steps above the original value, and the four steps below gave the values displayed in table 3. Figure 2 and 3 display a graphical representation of the data that is shown in table 3.

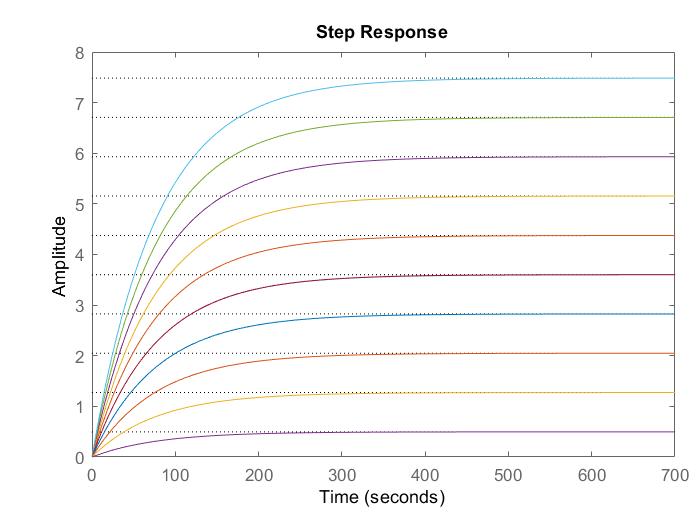


Figure 2: Step response from step change of heater constant

Table 3: In the table below are the results of the test for the variables of interest.

| Value of k | Color code | Rise time (seconds) | Peak time  (seconds) | Peak Amplitude (hertz) | Settling time  (seconds) |
| --- | --- | --- | --- | --- | --- |
| 0.0064 | Purple | 170.4692 | 818.2681 | .4696 | 303.5440 |
| .0164 | Bright Yellow | 170.5750 | 818.7764 | 1.2733 | 303.7325 |
| .0264 | Dark Orange | 170.5750 | 818.7764 | 2.0496 | 303.7325 |
| .0364 | Blue | 170.5750 | 818.7764 | 2.8260 | 303.7325 |
| .0464 | Maroon | 170.5750 | 818.7764 | 3.6024 | 303.7325 |
| .0564 | Orange | 170.4692 | 818.2681 | 4.3760 | 303.5440 |
| .0664 | Yellow | 170.5750 | 818.7764 | 5.1551 | 303.7325 |
| .0764 | Light Purple | 170.5750 | 818.7764 | 5.9315 | 303.7325 |
| .0864 | Green | 170.5750 | 818.7764 | 6.7079 | 303.7325 |
| .0964 | Light Blue | 170.5750 | 818.7764 | 7.4843 | 303.7325 |

From the data above it is clear that changing the value of k has little effect on the values of the rise, peak, and setting times, but has a positive correlation with peak amplitude. Figure 3 shows how changing the heater constant affects the poles and zeros of the system.

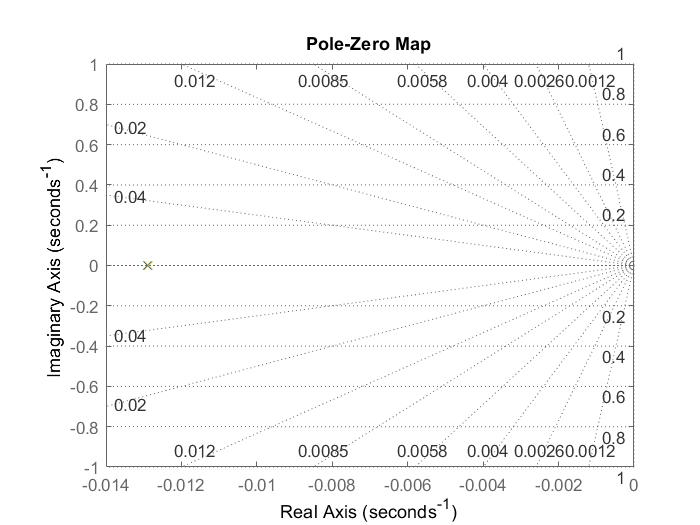


Figure 3: poles and zeros of function with regard to change in heater constant

Figure 3 shows that there is only one pole and no zeros associated with this system even with varying k values; the pole remained in one location, and was not affected. This is to be expected because the heater constant can only be set to zero if it is equal to zero, and the denominator is unchanging as k changes.

The figures and graphs and above indicate that the control system is stable. This can be determined due to the lack of oscillation in figure 2 and the pole being a negative value on the real axis shown in figure 3, also table 3 shows increase in only amplitude as the heater constant grows larger. This allows for the determination that the change in heater constant should only be altered if a larger output is desired, changing the heater constant will not increase or decrease time to reach a desired temperature, how long it will take for that temperature to become steady, or the time taken to reach maximum temperature.

The next test that was conducted was to vary the value of R, the resistance value that was given from the Hondianto et al. text and is displayed in the introduction in table 1 as 77.58 ohms. This value was varied from 50 to 100 in intervals of 5, excluding the value 75 which was replaced by the original value. Figure 4 below shows the curves associated with the step response trials.

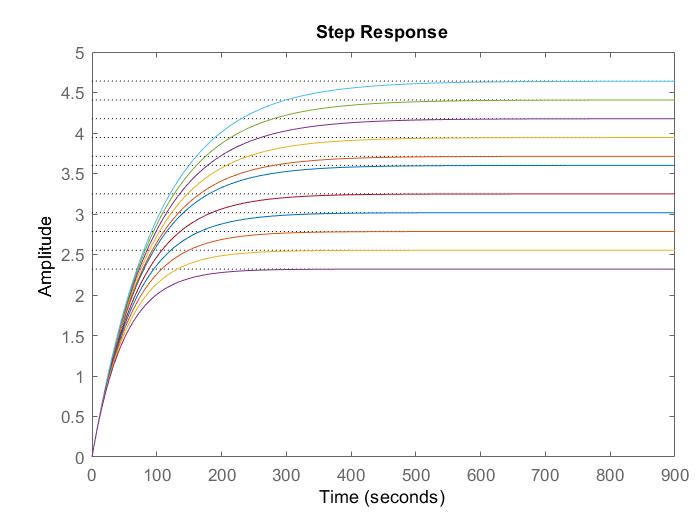


Figure 4: Step response due to step changes in resistance

In table 4 the data from figure 4 is shown, as well as a color legend, and the variables of interest to the test.

Table 4: Data from step response due to step changes in resistance

| Value of R (ohms) | Color code | Rise time (seconds) | Peak time  (seconds) | Peak Amplitude (hertz) | Settling time  (seconds) |
| --- | --- | --- | --- | --- | --- |
| 50 | Light purple | 109.8503 | 27.2920 | 2.3199 | 195.6037 |
| 55 | Bright yellow | 120.8354 | 580.0212 | 2.5519 | 215.1641 |
| 60 | Bright orange | 131.8204 | 632.7504 | 2.7839 | 234.7245 |
| 65 | Blue | 142.8054 | 685.4796 | 3.0159 | 254.2848 |
| 70 | Maroon | 153.7905 | 738.2088 | 3.2479 | 273.8452 |
| 77.58 | Dark blue | 170.4480 | 818.1666 | 3.5997 | 303.5063 |
| 80 | Orange | 175.7605 | 843.6672 | 3.7119 | 312.9660 |
| 85 | Yellow | 186.7456 | 896.3964 | 3.9439 | 332.5263 |
| 90 | Dark purple | 197.7306 | 949.1256 | 4.1759 | 352.0867 |
| 95 | Green | 208.7156 | 1.0019e+03 | 4.4079 | 371.6471 |
| 100 | Light blue | 219.7006 | 1.0546e+03 | 4.6399 | 391.2074 |

From table 4 it is apparent that changing the value of R has an effect on every variable of interest. Changing the resistance value has a positive correlation with rise, peak, and setting time as well as peak amplitude. Figure 5 shown below indicates how the poles and zeros change as the equation was altered.

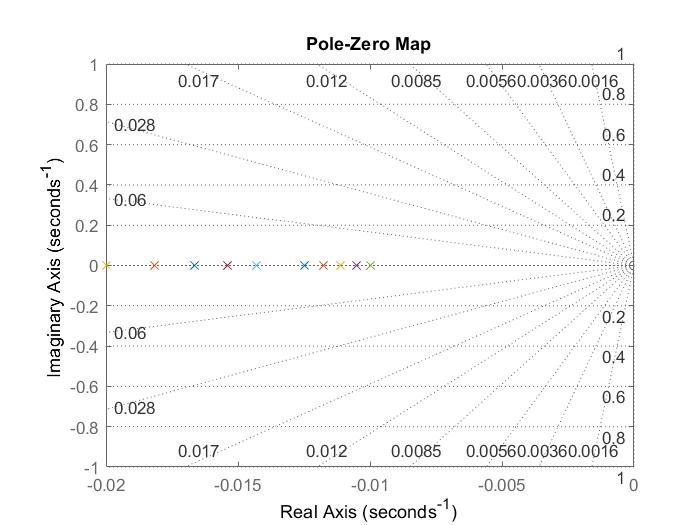


Figure 5: Poles and zeros of function with regard to changes in resistance

In figure 5 there are no zeros and a different singular pole for each resistance value. This is due to the fact that the resistance is a value whose inverse is in the denominator. When changed the value makes the denominator equal to zero at different values of s giving new pole values for each resistance value.

From these findings we can determine that the control system is stable. This can be determined due to the lack of oscillation in figure 4 and all the poles being negative and on the real axis shown in figure 5, also table 4 shows increases in all values as the resistance grows larger. This allows for the determination that the resistance should not be altered unless a larger output, and larger rise, peak, and/or setting time are required. This shows that resistance value has a large effect on the output on the system, and as stated above in less a complete change in all output value is desired changing the resistance value should not be changed.

The value of capacitance was then changed to see the effects that occured to the system. This was done by starting at the original value of 1 F and increasing by 5 F until reaching 25 F giveing 5 values above the original value, and 4 values below the original value. to obtain the four below the original value, the original value was halved, and then halved again, until four values were obtained. The data from these trials are displayed in figures 6 and 7 and table 5 shown below.

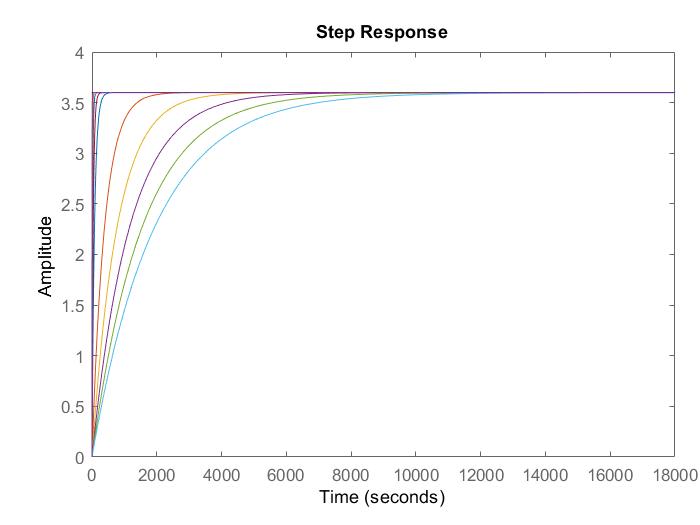
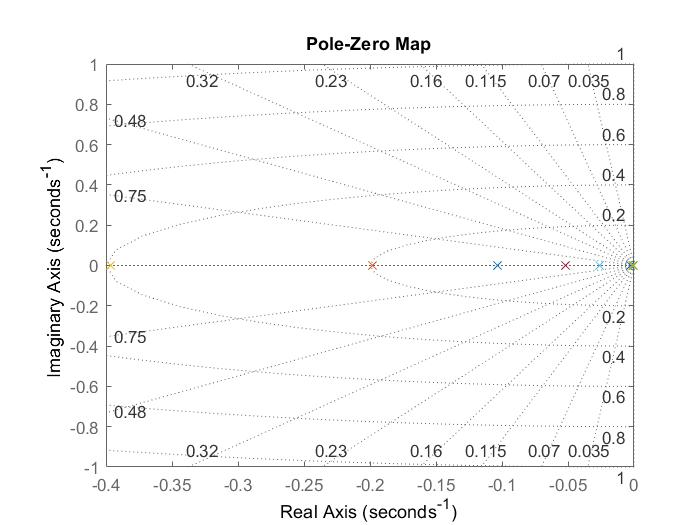
  
Figure 6: Step response due to step changes in capacitance

Figure 6 shows that capacitance has a drastic effect on the setting and rise time of the function. The data given in table 5 shows the values of the rise and settling times trends.

Table 5: Data from step response due to step changes in capacitance

| Value of c (Ferade) | Color code | Rise time (seconds) | Peak time  (seconds) | Peak Amplitude (hertz) | Settling time  (seconds) |
| --- | --- | --- | --- | --- | --- |
| .0325 | N/A | 5.5375 | 26.5805 | 3.5984 | 9.8603 |
| .065 | N/A | 11.0750 | 53.1610 | 3.5984 | 19.7206 |
| .125 | N/A | 21.2981 | 102.2327 | 3.5984 | 37.9242 |
| .25 | Dark blue | 42.5961 | 204.4655 | 3.5984 | 75.8483 |
| .5 | Maroon | 85.1923 | 408.9310 | 3.5984 | 151.6966 |
| 1 | Blue | 170.4480 | 818.1666 | 3.5997 | 303.5063 |
| 5 | Orange | 852.2400 | 4.0908e+03 | 3.5997 | 1.5175e+03 |
| 10 | Yellow | 1.7045e+03 | 8.1817e+03 | 3.5997 | 3.0351e+03 |
| 15 | Purple | 2.5567e+03 | 1.2272e+04 | 3.5997 | 4.5526e+03 |
| 20 | Green | 3.4077e+03 | 1.6357e+04 | 3.5984 | 6.0679e+03 |
| 25 | Light blue | 4.2596e+03 | 2.0447e+04 | 3.5984 | 7.5848e+03 |

In table 5 we can see that change in capacitance has an effect on rise, peak, and setting time, but no effect on peak amplitude. This is due to the capacitance value being multiplied by s meaning that as the coefficient value increases to infinity s also increases to infinity, so the lower the value of C the slower the response. In figure 7 below the effect of these changes on the poles and zeros are examined.

  
Figure 7: Poles and zeros of function with regard to changes in capacitance

The changing of capacitance does not have an effect on the zeros of the function, but does affect the poles. This is due to similar conditions to those of resistance mentioned above. However, unlike the resistance values the poles are further apart and a magnitude in size larger. This means that changing the capacitance values changes the overall response more than changing the resistance value.

The traits and trends of the figures and tables above show that the control system is still stable. This can be determined due to the lack of oscillation in figure 6 and all the poles being negative and on the real axis shown in figure 7, also table 5 shows increases in all values except peak amplitude as the capacitance grows larger. This allows for the determination that the capacitance should be altered if the same output is desired but at faster settling and rise time is required.

The penultimate test was to add extra poles to the equation to see how this changes the behavior of the function. In this test values of poles proceeded from -1 to -10, by multiplying the denominator by values of (s+x) with x ranging from 1 to 10. The results of this are shown in figures 8 and 9 and table 6.

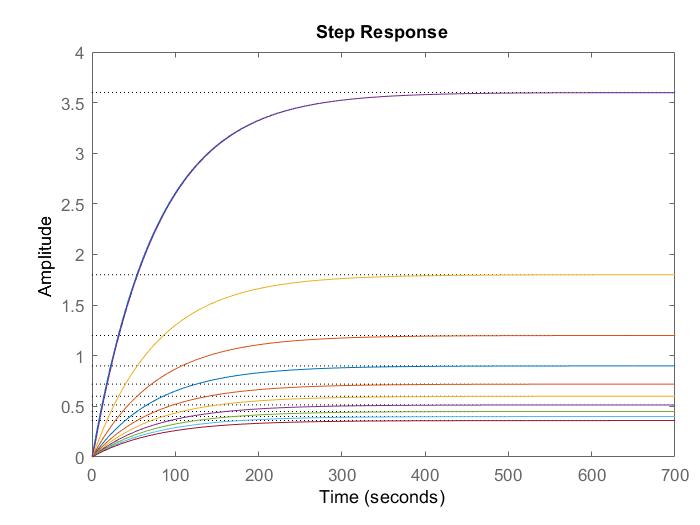


Figure 8: Step response due to step changes in added pole value

In figure 8 we see that the addition of poles clearly has an effect on the peak amplitude, but it is unclear to what extent. Table 6 gives more accurate and readable data in this regard.

Table 6: Data from step response due to step changes in added pole value

| Adding second pole | Color code | Rise time (seconds) | Peak time  (seconds) | Peak Amplitude (hertz) | Settling time  (seconds) |
| --- | --- | --- | --- | --- | --- |
| No pole added | N/A | 170.4480 | 818.1666 | 3.5997 | 303.5063 |
| (s+1) | Maroon | 170.4627 | 568.0747 | 3.5974 | 304.5249 |
| (s+2) | Light blue | 170.4752 | 818.2648 | 1.8001 | 304.0471 |
| (s+3) | Green | 170.5086 | 818.4360 | 1.2003 | 303.9396 |
| (s+4) | Purple | 170.4510 | 818.1646 | 0.8999 | 303.7535 |
| (s+5) | Yellow | 170.4504 | 818.1650 | 0.7199 | 303.7023 |
| (s+6) | Dark orange | 170.4712 | 818.2670 | 0.6000 | 303.7068 |
| (s+7) | Blue | 170.4369 | 818.1041 | 0.5142 | 303.6230 |
| (s+8) | Orange | 170.4468 | 818.1529 | 0.4500 | 303.6236 |
| (s+9) | Bright yellow | 170.4587 | 818.2109 | 0.4000 | 303.6315 |
| (s+10) | Dark blue | 170.4492 | 818.4360 | 0.3600 | 303.6039 |

From table 6 it becomes clear that the more negative the pole value the less the amplitude, but there is very little effect on the rise, peak and setting times with respect to different pole values. Figure 9 shows the pole values graphically.

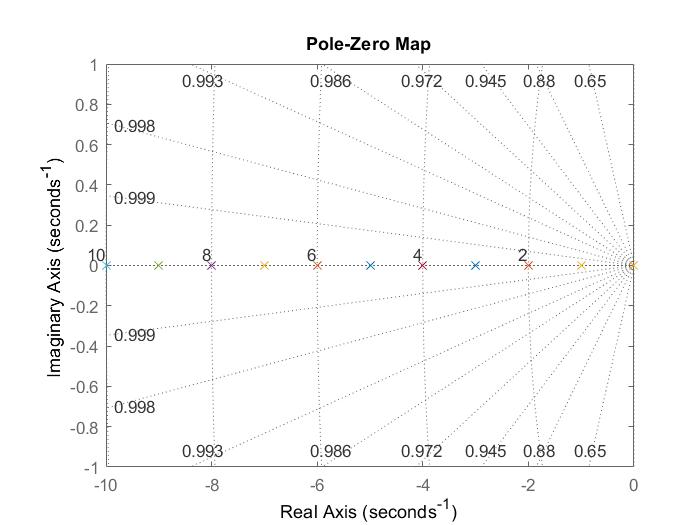


Figure 9: Poles and zeros of function with regard to changes in added pole value

From figure 9 we can see that adding a pole value has no effect on the zero values. It appears that there is no noticeable effect on the original pole value that can be seen right above zero on the real axis. We can also see all the values of all the added poles from -1 through -10.

In this test the data shows that the control system is still stable. This can be determined due to the lack of oscillation in figure 8 and all the poles being negative and on the real axis shown in figure 9, also table 6 shows decrease in peak amplitude as the poles become more negative. This allows for the determination that if poles can be added then they should only be added if you wish to alter the peak amplitude, without causing significant changes to the rise, peak, and setting times. Adding negative poles will not increase or decrease time to reach or stabilize or desired peak temperature,

In the final test various zeros were added to the numerator, in order to understand how the function would react to the presence of zeros. The zeros ranged in value from -1 to -10 similarly to the poles experiment. The results are shown in figures 10, and 11 and in table 7.

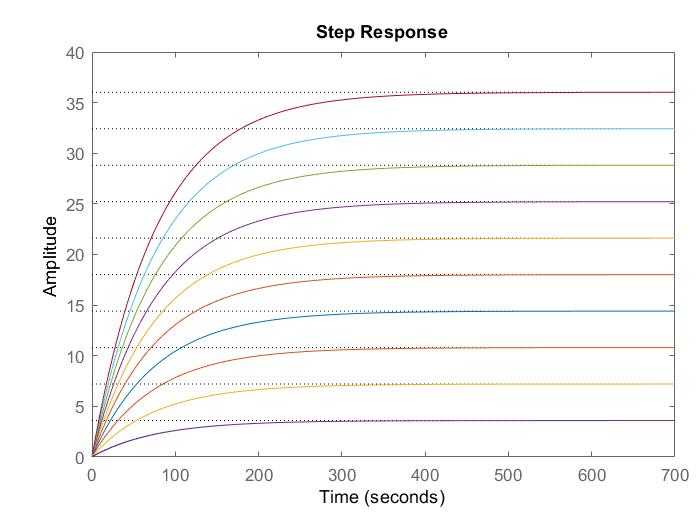


Figure 10: Step response due to step changes in added zero value

In figure 10 it can be seen that different zero values affect the amplitude of the output with little to no effect on the rise, peak, and setting times. For further investigation of the data table 7 below gives more detailed data.

Table 7: Data from step response due to step changes in added zero value

| Adding a zero | Color code | Rise time (seconds) | Peak time  (seconds) | Peak Amplitude (hertz) | Settling time  (seconds) |
| --- | --- | --- | --- | --- | --- |
| No zero added | N/A | 170.4480 | 818.1666 | 3.5997 | 303.5063 |
| (s+1) | Dark purple | 170.5750 | 818.7764 | 3.6024 | 303.7325 |
| (s+2) | Bright yellow | 170.5750 | 818.7764 | 7.2048 | 303.7325 |
| (s+3) | Orange | 170.5750 | 818.7764 | 10.8072 | 303.7325 |
| (s+4) | Blue | 170.5750 | 818.7764 | 14.4096 | 303.7325 |
| (s+5) | Dark Orange | 170.5750 | 818.7764 | 18.0119 | 303.7325 |
| (s+6) | Yellow | 170.5750 | 818.7764 | 21.6143 | 303.7325 |
| (s+7) | Purple | 170.5750 | 818.7764 | 25.2167 | 303.7325 |
| (s+8) | Green | 170.5750 | 818.7764 | 28.8191 | 303.7325 |
| (s+9) | Light blue | 170.5750 | 818.7764 | 32.4215 | 303.7325 |
| (s+10) | Maroon | 170.5750 | 818.7764 | 36.0239 | 303.7325 |

From table 7 it is clear that adding a zero of specified value between -1 and -10 only affects the peak amplitude, and that as the zero becomes more negative, the amplitude increases. Figure 11 graphically shows the poles and zeros of this system.

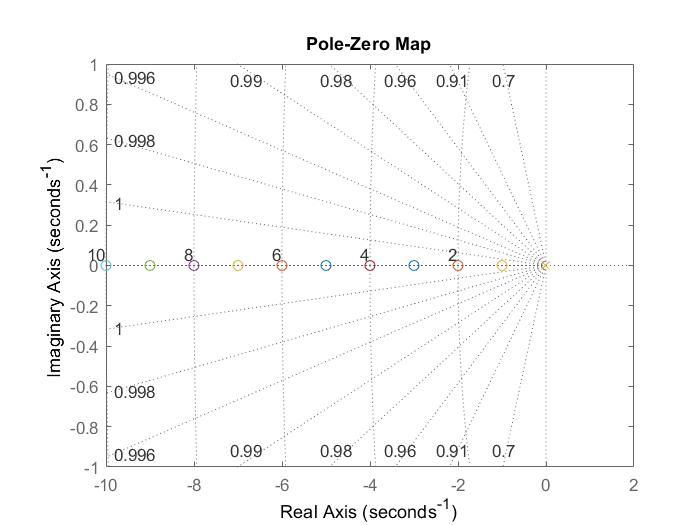
****

Figure 11: Poles and zeros of function with regard to changes in added zero value

From figure 11 it is clear that adding these zero values has no noticeable effect on the original pole value. This also lists the zero values on the real axis.

In this test the data shows that the control system is still stable. This can be determined due to the lack of oscillation in figure 10 and all the poles being negative and on the real axis shown in figure 11, also table 7 shows increase in peak amplitude as the zeros become more negative. This allows for the determination that if zeros can be added then they should only be added if you wish to alter the peak amplitude, without causing significant changes to the rise, peak, and setting times. Similarly to adding poles, Adding negative zeros will not increase or decrease time to reach or stabilize or desired peak temperature,

**Conclusion**

In conclusion, it appears that this portion of our chosen transfer function is stable at every condition presented in this paper. Each of the tests had a novel and notable effect on the output of the transfer function. If the k value is varied, the amplitude varies with it. If the R value is changed, all variables of interest are changed. If the C value is increased or decreased it has a positive correlation with rise, peak and setting time without changing peak amplitude. If negative poles or zeros are added they affect peak amplitude without affecting rise, peak, and setting time. More negative poles decrease amplitude, more negative zeros increase amplitude.

**Recommendations**

It is recommended that further investigation be done with regard to whether a pole or zero can be realistically inserted into the system using mechanical or computational additions to the physical system. Another Recommendation is that no further research be directed towards increasing the resistance of the system because the positives of this have equal trade offs, if you wished for faster performance, you would have to lose temperature. It would be useful to examine other parts of the overall transfer function to check for their stability with regard to their variables. It is also recommended that persons with better computational power, and more time to investigate the full transfer function rather than a small portion at a time.

**References**

[1] Hondianto T. and Susanto E. and Wibowo A., 2016, “Model Driven PID Controller in Water Heater System.” *International Journal of Electrical and Computer Engineering*, 6. 1673. 10.11591/ijece.v6i4.pp1673-1680, from <https://www.researchgate.net/publication/336884650_Model_Driven_PID_Controller_in_Water_Heater_System>

**[2] “Electrical Water Heaters.”** *Electrical Water Heaters | The ASHI Reporter | Inspection News & Views from the American Society of Home Inspectors*, www.ashireporter.org/HomeInspection/Articles/Electrical-Water-Heaters/15053.

**Appendix A:**

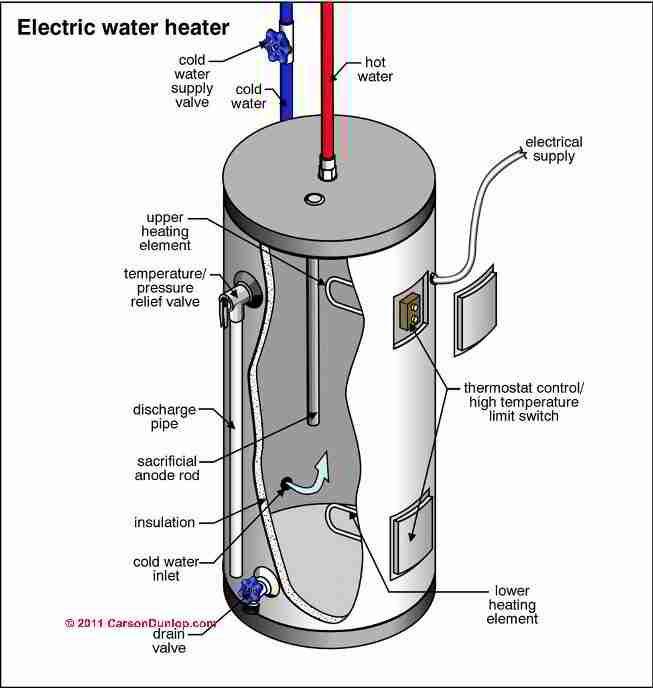
****

figure 12: Tepid water heater schematic

**Appendix B:**

**Programming in Matlab to model Transfer Function**

syms s;

k=.0464;

k1=.0564;

k2=.0664;

k3=.0764;

k4=.0864;

k5=.0964;

k6=.0464;

k7=.0364;

k8=.0264;

k9=.0164;

k10=.0064;

r=77.58;

r1=80;

r2=85;

r3=90;

r4=95;

r5=100;

r6=70;

r7=65;

r8=60;

r9=55;

r10=50;

c=1;

c1=5;

c2=10;

c3=15;

c4=20;

c5=25;

c6=.5;

c7=.25;

c8=.125;

c9=.065;

c10=.0325;

tau=30;

kb=k\*r;

tk=c\*r;

K=2.37;

t=33.02;

L=34.4;

a=0;

lamda=1;

p(s)=(k)/(c\*s+(1/r));

p(s)=(k1)/(c\*s+(1/r));

p(s)=(k2)/(c\*s+(1/r));

p(s)=(k3)/(c\*s+(1/r));

p(s)=(k4)/(c\*s+(1/r));

p(s)=(k5)/(c\*s+(1/r));

p(s)=(k6)/(c\*s+(1/r));

p(s)=(k7)/(c\*s+(1/r));

p(s)=(k8)/(c\*s+(1/r));

p(s)=(k9)/(c\*s+(1/r));

p(s)=(k10)/(c\*s+(1/r));

num=[29];

num1=[141];

num2=[83];

num3=[191];

num4=[54];

num5=[241];

num6=[29];

num7=[91];

num8=[33];

num9=[41];

num10=[8];

den=[625 8.056];

den1=[2500 32.22];

den2=[1250 16.1];

den3=[2500 32.2];

den4=[625 8.05];

den5=[2500 32.2];

den6=[625 8.05];

den7=[2500 32.2];

den8=[1250 16.1];

den9=[2500 32.2];

den10=[125 1.61];

tfo=tf(num,den);

tf1=tf(num1,den1);

tf2=tf(num2,den2);

tf3=tf(num3,den3);

tf4=tf(num4,den4);

tf5=tf(num5,den5);

tf6=tf(num6,den6);

tf7=tf(num7,den7);

tf8=tf(num8,den8);

tf9=tf(num9,den9);

tf10=tf(num10,den10);

linearSystemAnalyzer(tfo,tf1,tf2,tf3,tf4,tf5,tf6,tf7,tf8,tf9,tf10)

disp("the below section covers the change in transfer function at the heater constant value value is changed")

disp(" ")

disp("step info original transfer function")

stepinfo(tfo)

disp(" ")

disp("step info k1")

stepinfo(tf1)

disp(" ")

disp("step info k2")

stepinfo(tf2)

disp(" ")

disp("step info k3")

stepinfo(tf3)

disp(" ")

disp("step info k4")

stepinfo(tf4)

disp(" ")

disp("step info k5")

stepinfo(tf5)

disp(" ")

disp("step info k6")

stepinfo(tf6)

disp(" ")

disp("step info k7")

stepinfo(tf7)

disp(" ")

disp("step info k8")

stepinfo(tf8)

disp(" ")

disp("step info k9")

stepinfo(tf9)

disp(" ")

disp("step info k10")

stepinfo(tf10)

p(s)=(k)/(c\*s+(1/r1));

p(s)=(k)/(c\*s+(1/r2));

p(s)=(k)/(c\*s+(1/r3));

p(s)=(k)/(c\*s+(1/r4));

p(s)=(k)/(c\*s+(1/r5));

p(s)=(k)/(c\*s+(1/r6));

p(s)=(k)/(c\*s+(1/r7));

p(s)=(k)/(c\*s+(1/r8));

p(s)=(k)/(c\*s+(1/r9));

p(s)=(k)/(c\*s+(1/r10));

num11=[29];

num12=[29];

num13=[29];

num14=[29];

num15=[29];

num16=[29];

num17=[29];

num18=[29];

num19=[29];

num20=[29];

den11=[625 1/80];

den12=[625 1/85];

den13=[625 1/90];

den14=[625 1/95];

den15=[625 1/100];

den16=[625 1/70];

den17=[625 1/65];

den18=[625 1/60];

den19=[625 1/55];

den20=[625 1/50];

tf11=tf(num11,den11);

tf12=tf(num12,den12);

tf13=tf(num13,den13);

tf14=tf(num14,den14);

tf15=tf(num15,den15);

tf16=tf(num16,den16);

tf17=tf(num17,den17);

tf18=tf(num18,den18);

tf19=tf(num19,den19);

tf20=tf(num20,den20);

linearSystemAnalyzer(tfo,tf11,tf12,tf13,tf14,tf15,tf16,tf17,tf18,tf19,tf20)

disp("the below section covers the change in transfer function at the resistance value is changed")

disp("step info original transfer function")

stepinfo(tfo)

disp(" ")

disp("step info r1")

stepinfo(tf11)

disp(" ")

disp("step info r2")

stepinfo(tf12)

disp(" ")

disp("step info r3")

stepinfo(tf13)

disp(" ")

disp("step info r4")

stepinfo(tf14)

disp(" ")

disp("step info r5")

stepinfo(tf15)

disp(" ")

disp("step info r6")

stepinfo(tf16)

disp(" ")

disp("step info r7")

stepinfo(tf17)

disp(" ")

disp("step info r8")

stepinfo(tf18)

disp(" ")

disp("step info r9")

stepinfo(tf19)

disp(" ")

disp("step info r10")

stepinfo(tf20)

p(s)=(k)/(c1\*s+(1/r));

p(s)=(k)/(c2\*s+(1/r));

p(s)=(k)/(c3\*s+(1/r));

p(s)=(k)/(c4\*s+(1/r));

p(s)=(k)/(c5\*s+(1/r));

p(s)=(k)/(c6\*s+(1/r));

p(s)=(k)/(c7\*s+(1/r));

p(s)=(k)/(c8\*s+(1/r));

p(s)=(k)/(c9\*s+(1/r));

p(s)=(k)/(c10\*s+(1/r));

num21=[29];

num22=[29];

num23=[29];

num24=[29];

num25=[29];

num26=[29];

num27=[29];

num28=[29];

num29=[29];

num30=[29];

den21=[3125 8.056];

den22=[6250 8.056];

den23=[9375 8.056];

den24=[12500 8.059];

den25=[15625 8.059];

den26=[312.5 8.059];

den27=[156.25 8.059];

den28=[78.125 8.059];

den29=[40.625 8.059];

den30=[20.3125 8.059];

tf21=tf(num21,den21);

tf22=tf(num22,den22);

tf23=tf(num23,den23);

tf24=tf(num24,den24);

tf25=tf(num25,den25);

tf26=tf(num26,den26);

tf27=tf(num27,den27);

tf28=tf(num28,den28);

tf29=tf(num29,den29);

tf30=tf(num30,den30);

linearSystemAnalyzer(tfo,tf21,tf22,tf23,tf24,tf25,tf26,tf27,tf28,tf29,tf30)

disp("the below section covers the change in transfer function at the capacitance value is changed")

disp("step info original transfer function")

stepinfo(tfo)

disp(" ")

disp("step info c1")

stepinfo(tf21)

disp(" ")

disp("step info c2")

stepinfo(tf22)

disp(" ")

disp("step info c3")

stepinfo(tf23)

disp(" ")

disp("step info c4")

stepinfo(tf24)

disp(" ")

disp("step info c5")

stepinfo(tf25)

disp(" ")

disp("step info c6")

stepinfo(tf26)

disp(" ")

disp("step info c7")

stepinfo(tf27)

disp(" ")

disp("step info c8")

stepinfo(tf28)

disp(" ")

disp("step info c9")

stepinfo(tf29)

disp(" ")

disp("step info c10")

stepinfo(tf30)

disp("the below section covers adding a second pole at changing values")

p(s)=(k)/(c\*s\*(s+5)+(1/r)\*(s+5));

p(s)=(k)/(c\*s\*(s+6)+(1/r)\*(s+6));

p(s)=(k)/(c\*s\*(s+7)+(1/r)\*(s+7));

p(s)=(k)/(c\*s\*(s+8)+(1/r)\*(s+8));

p(s)=(k)/(c\*s\*(s+9)+(1/r)\*(s+9));

p(s)=(k)/(c\*s\*(s+10)+(1/r)\*(s+10));

p(s)=(k)/(c\*s\*(s+4)+(1/r)\*(s+4));

p(s)=(k)/(c\*s\*(s+3)+(1/r)\*(s+3));

p(s)=(k)/(c\*s\*(s+2)+(1/r)\*(s+2));

p(s)=(k)/(c\*s\*(s+1)+(1/r)\*(s+1));

num31=[29];

num32=[29];

num33=[29];

num34=[29];

num35=[29];

num36=[29];

num37=[29];

num38=[29];

num39=[29];

num40=[29];

den31=[625 3133.05 40.28];

den32=[625 3758.05 48.33];

den33=[625 4383.056 56.3963];

den34=[625 5008.05 64.449];

den35=[625 5633.05 72.5];

den36=[625 6258.05 80.56];

den37=[625 2508.05 32.224];

den38=[625 1883.05 24.16];

den39=[625 1258.05 16.11];

den40=[625 633.059 8.056];

tf31=tf(num31,den31);

tf32=tf(num32,den32);

tf33=tf(num33,den33);

tf34=tf(num34,den34);

tf35=tf(num35,den35);

tf36=tf(num36,den36);

tf37=tf(num37,den37);

tf38=tf(num38,den38);

tf39=tf(num39,den39);

tf40=tf(num40,den40);

linearSystemAnalyzer(tfo,tf31,tf32,tf33,tf34,tf35,tf36,tf37,tf38,tf39,tf40)

disp("step info original transfer function")

stepinfo(tfo)

disp(" ")

disp("step info adding pole (s+5)")

stepinfo(tf31)

disp(" ")

disp("step info adding pole (s+6)")

stepinfo(tf32)

disp(" ")

disp("step info adding pole (s+7)")

stepinfo(tf33)

disp(" ")

disp("step info adding pole (s+8)")

stepinfo(tf34)

disp(" ")

disp("step info adding pole (s+9)")

stepinfo(tf35)

disp(" ")

disp("step info adding pole (s+10)")

stepinfo(tf36)

disp(" ")

disp("step info adding pole (s+4)")

stepinfo(tf37)

disp(" ")

disp("step info adding pole (s+3)")

stepinfo(tf38)

disp(" ")

disp("step info adding pole (s+2)")

stepinfo(tf39)

disp(" ")

disp("step info adding pole (s+1)")

stepinfo(tf40)

disp("the below section covers adding a zero at changing values")

p(s)=((k)\*(s+5))/(c\*s+(1/r));

p(s)=((k)\*(s+6))/(c\*s+(1/r));

p(s)=((k)\*(s+7))/(c\*s+(1/r));

p(s)=((k)\*(s+8))/(c\*s+(1/r));

p(s)=((k)\*(s+9))/(c\*s+(1/r));

p(s)=((k)\*(s+10))/(c\*s+(1/r));

p(s)=((k)\*(s+4))/(c\*s+(1/r));

p(s)=((k)\*(s+3))/(c\*s+(1/r));

p(s)=((k)\*(s+2))/(c\*s+(1/r));

p(s)=((k)\*(s+1))/(c\*s+(1/r));

den41=[1 .01288];

den42=[1 .01288];

den43=[1 .01288];

den44=[1 .01288];

den45=[1 .01288];

den46=[1 .01288];

den47=[1 .01288];

den48=[1 .01288];

den49=[1 .01288];

den50=[1 .01288];

num41=[.0464 .232];

num42=[.0464 .2784];

num43=[.0464 .3248];

num44=[.0464 .3712];

num45=[.0464 .4176];

num46=[.0464 .464];

num47=[.0464 .1856];

num48=[.0464 .1392];

num49=[.0464 .0928];

num50=[.0464 .0464];

tf41=tf(num41,den41);

tf42=tf(num42,den42);

tf43=tf(num43,den43);

tf44=tf(num44,den44);

tf45=tf(num45,den45);

tf46=tf(num46,den46);

tf47=tf(num47,den47);

tf48=tf(num48,den48);

tf49=tf(num49,den49);

tf50=tf(num50,den50);

linearSystemAnalyzer(tfo,tf41,tf42,tf43,tf44,tf45,tf46,tf47,tf48,tf49,tf50)

disp("step info original transfer function")

stepinfo(tfo)

disp(" ")

disp("step info adding zero (s+5)")

stepinfo(tf41)

disp(" ")

disp("step info adding zero (s+6)")

stepinfo(tf42)

disp(" ")

disp("step info adding zero (s+7)")

stepinfo(tf43)

disp(" ")

disp("step info adding zero (s+8)")

stepinfo(tf44)

disp(" ")

disp("step info adding zero (s+9)")

stepinfo(tf45)

disp(" ")

disp("step info adding zero (s+10)")

stepinfo(tf46)

disp(" ")

disp("step info adding zero (s+4)")

stepinfo(tf47)

disp(" ")

disp("step info adding zero (s+3)")

stepinfo(tf48)

disp(" ")

disp("step info adding zero (s+2)")

stepinfo(tf49)

disp(" ")

disp("step info adding zero (s+1)")

stepinfo(tf50)

**Resulting output to code**

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

step info k1

ans =

struct with fields:

RiseTime: 170.4692

SettlingTime: 303.5440

SettlingMin: 3.9582

SettlingMax: 4.3760

Overshoot: 0

Undershoot: 0

Peak: 4.3760

PeakTime: 818.2681

step info k2

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 4.6630

SettlingMax: 5.1551

Overshoot: 0

Undershoot: 0

Peak: 5.1551

PeakTime: 818.7764

step info k3

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 5.3652

SettlingMax: 5.9315

Overshoot: 0

Undershoot: 0

Peak: 5.9315

PeakTime: 818.7764

step info k4

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 6.0675

SettlingMax: 6.7079

Overshoot: 0

Undershoot: 0

Peak: 6.7079

PeakTime: 818.7764

step info k5

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 6.7697

SettlingMax: 7.4843

Overshoot: 0

Undershoot: 0

Peak: 7.4843

PeakTime: 818.7764

step info k6

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 3.2584

SettlingMax: 3.6024

Overshoot: 0

Undershoot: 0

Peak: 3.6024

PeakTime: 818.7764

step info k7

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 2.5562

SettlingMax: 2.8260

Overshoot: 0

Undershoot: 0

Peak: 2.8260

PeakTime: 818.7764

step info k8

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 1.8539

SettlingMax: 2.0496

Overshoot: 0

Undershoot: 0

Peak: 2.0496

PeakTime: 818.7764

step info k9

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 1.1517

SettlingMax: 1.2733

Overshoot: 0

Undershoot: 0

Peak: 1.2733

PeakTime: 818.7764

step info k10

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 4.4944

SettlingMax: 4.9688

Overshoot: 0

Undershoot: 0

Peak: 4.9688

PeakTime: 818.7764

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

the below section covers the change in transfer function at the resistance value is changed

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

step info r1

ans =

struct with fields:

RiseTime: 175.7605

SettlingTime: 312.9660

SettlingMin: 3.3575

SettlingMax: 3.7119

Overshoot: 0

Undershoot: 0

Peak: 3.7119

PeakTime: 843.6672

step info r2

ans =

struct with fields:

RiseTime: 186.7456

SettlingTime: 332.5263

SettlingMin: 3.5674

SettlingMax: 3.9439

Overshoot: 0

Undershoot: 0

Peak: 3.9439

PeakTime: 896.3964

step info r3

ans =

struct with fields:

RiseTime: 197.7306

SettlingTime: 352.0867

SettlingMin: 3.7772

SettlingMax: 4.1759

Overshoot: 0

Undershoot: 0

Peak: 4.1759

PeakTime: 949.1256

step info r4

ans =

struct with fields:

RiseTime: 208.7156

SettlingTime: 371.6471

SettlingMin: 3.9870

SettlingMax: 4.4079

Overshoot: 0

Undershoot: 0

Peak: 4.4079

PeakTime: 1.0019e+03

step info r5

ans =

struct with fields:

RiseTime: 219.7006

SettlingTime: 391.2074

SettlingMin: 4.1969

SettlingMax: 4.6399

Overshoot: 0

Undershoot: 0

Peak: 4.6399

PeakTime: 1.0546e+03

step info r6

ans =

struct with fields:

RiseTime: 153.7905

SettlingTime: 273.8452

SettlingMin: 2.9378

SettlingMax: 3.2479

Overshoot: 0

Undershoot: 0

Peak: 3.2479

PeakTime: 738.2088

step info r7

ans =

struct with fields:

RiseTime: 142.8054

SettlingTime: 254.2848

SettlingMin: 2.7280

SettlingMax: 3.0159

Overshoot: 0

Undershoot: 0

Peak: 3.0159

PeakTime: 685.4796

step info r8

ans =

struct with fields:

RiseTime: 131.8204

SettlingTime: 234.7245

SettlingMin: 2.5181

SettlingMax: 2.7839

Overshoot: 0

Undershoot: 0

Peak: 2.7839

PeakTime: 632.7504

step info r9

ans =

struct with fields:

RiseTime: 120.8354

SettlingTime: 215.1641

SettlingMin: 2.3083

SettlingMax: 2.5519

Overshoot: 0

Undershoot: 0

Peak: 2.5519

PeakTime: 580.0212

step info r10

ans =

struct with fields:

RiseTime: 109.8503

SettlingTime: 195.6037

SettlingMin: 2.0984

SettlingMax: 2.3199

Overshoot: 0

Undershoot: 0

Peak: 2.3199

PeakTime: 527.2920

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

step info c1

ans =

struct with fields:

RiseTime: 852.2400

SettlingTime: 1.5175e+03

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 4.0908e+03

step info c2

ans =

struct with fields:

RiseTime: 1.7045e+03

SettlingTime: 3.0351e+03

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 8.1817e+03

step info c3

ans =

struct with fields:

RiseTime: 2.5567e+03

SettlingTime: 4.5526e+03

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 1.2272e+04

step info c4

ans =

struct with fields:

RiseTime: 3.4077e+03

SettlingTime: 6.0679e+03

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 1.6357e+04

step info c5

ans =

struct with fields:

RiseTime: 4.2596e+03

SettlingTime: 7.5848e+03

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 2.0447e+04

step info c6

ans =

struct with fields:

RiseTime: 85.1923

SettlingTime: 151.6966

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 408.9310

step info c7

ans =

struct with fields:

RiseTime: 42.5961

SettlingTime: 75.8483

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 204.4655

step info c8

ans =

struct with fields:

RiseTime: 21.2981

SettlingTime: 37.9242

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 102.2327

step info c9

ans =

struct with fields:

RiseTime: 11.0750

SettlingTime: 19.7206

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 53.1610

step info c10

ans =

struct with fields:

RiseTime: 5.5375

SettlingTime: 9.8603

SettlingMin: 3.2548

SettlingMax: 3.5984

Overshoot: 0

Undershoot: 0

Peak: 3.5984

PeakTime: 26.5805

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

step info adding pole (s+5)

ans =

struct with fields:

RiseTime: 170.4504

SettlingTime: 303.7023

SettlingMin: 0.6510

SettlingMax: 0.7199

Overshoot: 0

Undershoot: 0

Peak: 0.7199

PeakTime: 818.1650

step info adding pole (s+6)

ans =

struct with fields:

RiseTime: 170.4712

SettlingTime: 303.7068

SettlingMin: 0.5426

SettlingMax: 0.6000

Overshoot: 0

Undershoot: 0

Peak: 0.6000

PeakTime: 818.2670

step info adding pole (s+7)

ans =

struct with fields:

RiseTime: 170.4369

SettlingTime: 303.6230

SettlingMin: 0.4650

SettlingMax: 0.5142

Overshoot: 0

Undershoot: 0

Peak: 0.5142

PeakTime: 818.1041

step info adding pole (s+8)

ans =

struct with fields:

RiseTime: 170.4468

SettlingTime: 303.6236

SettlingMin: 0.4069

SettlingMax: 0.4500

Overshoot: 0

Undershoot: 0

Peak: 0.4500

PeakTime: 818.1529

step info adding pole (s+9)

ans =

struct with fields:

RiseTime: 170.4587

SettlingTime: 303.6315

SettlingMin: 0.3617

SettlingMax: 0.4000

Overshoot: 0

Undershoot: 0

Peak: 0.4000

PeakTime: 818.2109

step info adding pole (s+10)

ans =

struct with fields:

RiseTime: 170.4492

SettlingTime: 303.6039

SettlingMin: 0.3256

SettlingMax: 0.3600

Overshoot: 0

Undershoot: 0

Peak: 0.3600

PeakTime: 818.1658

step info adding pole (s+4)

ans =

struct with fields:

RiseTime: 170.4510

SettlingTime: 303.7535

SettlingMin: 0.8137

SettlingMax: 0.8999

Overshoot: 0

Undershoot: 0

Peak: 0.8999

PeakTime: 818.1646

step info adding pole (s+3)

ans =

struct with fields:

RiseTime: 170.5086

SettlingTime: 303.9396

SettlingMin: 1.0852

SettlingMax: 1.2003

Overshoot: 0

Undershoot: 0

Peak: 1.2003

PeakTime: 818.4360

step info adding pole (s+2)

ans =

struct with fields:

RiseTime: 170.4752

SettlingTime: 304.0471

SettlingMin: 1.6271

SettlingMax: 1.8001

Overshoot: 0

Undershoot: 0

Peak: 1.8001

PeakTime: 818.2648

step info adding pole (s+1)

ans =

struct with fields:

RiseTime: 170.4627

SettlingTime: 304.5249

SettlingMin: 3.2515

SettlingMax: 3.5974

Overshoot: 0

Undershoot: 0

Peak: 3.5974

PeakTime: 568.0747

step info original transfer function

ans =

struct with fields:

RiseTime: 170.4480

SettlingTime: 303.5063

SettlingMin: 3.2560

SettlingMax: 3.5997

Overshoot: 0

Undershoot: 0

Peak: 3.5997

PeakTime: 818.1666

step info adding zero (s+5)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 16.2967

SettlingMax: 18.0119

Overshoot: 0

Undershoot: 0

Peak: 18.0119

PeakTime: 818.7764

step info adding zero (s+6)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 19.5551

SettlingMax: 21.6143

Overshoot: 0

Undershoot: 0

Peak: 21.6143

PeakTime: 818.7764

step info adding zero (s+7)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 22.8136

SettlingMax: 25.2167

Overshoot: 0

Undershoot: 0

Peak: 25.2167

PeakTime: 818.7764

step info adding zero (s+8)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 26.0720

SettlingMax: 28.8191

Overshoot: 0

Undershoot: 0

Peak: 28.8191

PeakTime: 818.7764

step info adding zero (s+9)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 29.3305

SettlingMax: 32.4215

Overshoot: 0

Undershoot: 0

Peak: 32.4215

PeakTime: 818.7764

step info adding zero (s+10)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 32.5889

SettlingMax: 36.0239

Overshoot: 0

Undershoot: 0

Peak: 36.0239

PeakTime: 818.7764

step info adding zero (s+4)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 13.0382

SettlingMax: 14.4096

Overshoot: 0

Undershoot: 0

Peak: 14.4096

PeakTime: 818.7764

step info adding zero (s+3)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 9.7798

SettlingMax: 10.8072

Overshoot: 0

Undershoot: 0

Peak: 10.8072

PeakTime: 818.7764

step info adding zero (s+2)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 6.5213

SettlingMax: 7.2048

Overshoot: 0

Undershoot: 0

Peak: 7.2048

PeakTime: 818.7764

step info adding zero (s+1)

ans =

struct with fields:

RiseTime: 170.5750

SettlingTime: 303.7325

SettlingMin: 3.2629

SettlingMax: 3.6024

Overshoot: 0

Undershoot: 0

Peak: 3.6024

PeakTime: 818.7764

pzmap(tf1,tf2,tf3,tf4,tf5,tf6,tf7,tf8,tf9,tf10)

%pzmap(tf11,tf12,tf13,tf14,tf15,tf16,tf17,tf18,tf19,tf20)

%pzmap(tf21,tf22,tf23,tf24,tf25,tf26,tf27,tf28,tf29,tf30)

%pzmap(tf31,tf32,tf33,tf34,tf35,tf36,tf37,tf38,tf39,tf40)

%pzmap(tf41,tf42,tf43,tf44,tf45,tf46,tf47,tf48,tf49,tf50)

grid on