3/4/2021

Investigating PID Control Behaviors in Simulations

Georgia Institute of Technology

Electronics II with Professor First

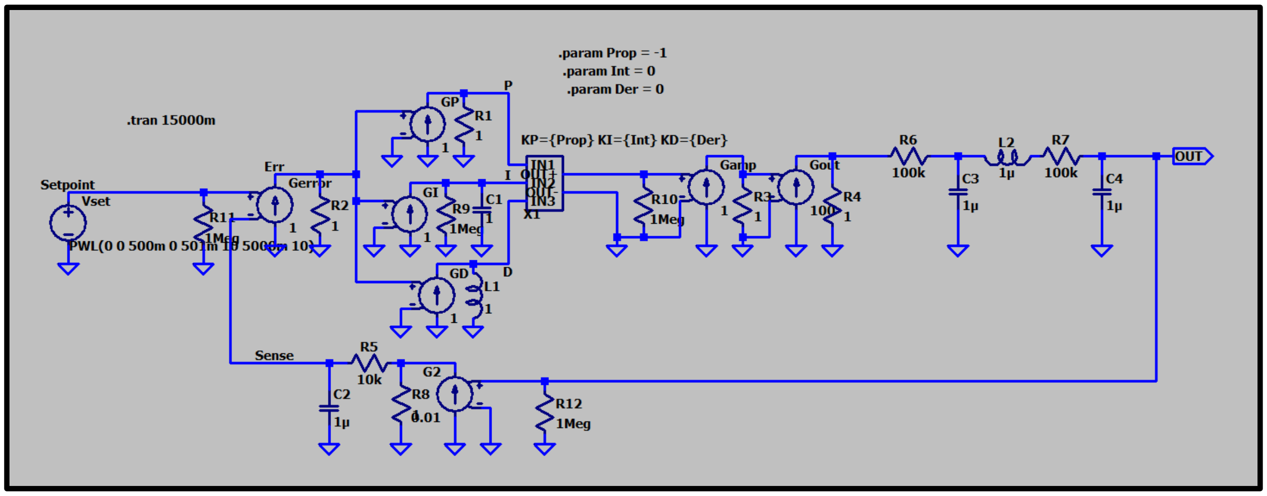
Robert Pierrard

INTRODUCTION

One of the most favorable types of feedback controllers across various industries is the PID controller. Its diverse application stems from each control process being independently tunable. In this way, different PID controllers can exhibit very different behaviors, and thus be applied to different systems. In reality, most PID controllers are designed precisely for their exact control application. The purpose of this lab is to develop an understanding of PID control by observing the effects of PID parameters through simulations. In particular, LTspice will be used to simulate a control loop in its entirety, from setpoint to output.

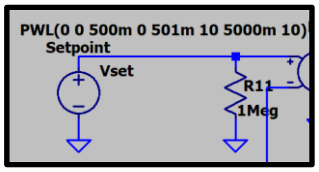
SIMULATION SETUP

Our simulation is conducted through LT Spice. By using such a simulation software, we can readily quantify the effects of modifying each PID parameter. The following circuit is a minimally modified version of a circuit provided by Professor First. It is used for the duration of this lab.



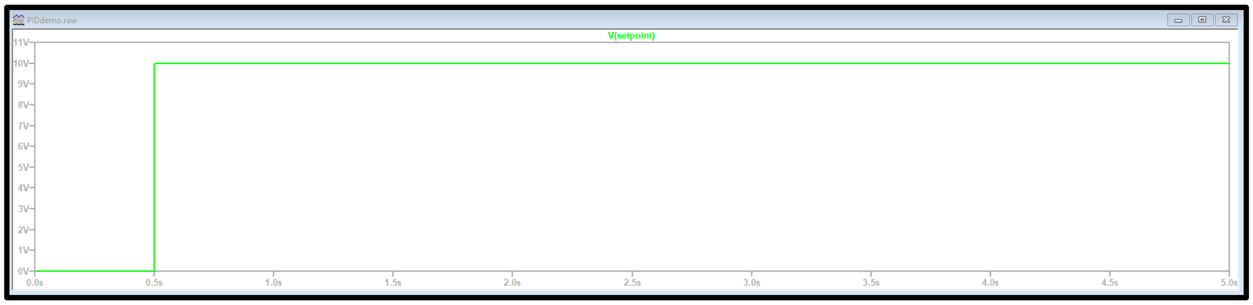
To understand the effects of the PID controls, we must first understand the structure of this circuit. We will now explore each subcircuit that is relevant for understanding PID control.

Setpoint Voltage

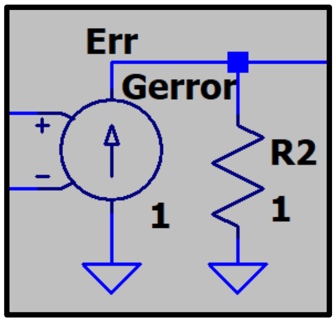


First, we have the setpoint voltage. This is the part of the circuit where we set our desired output. It is analogous to setting the temperature on a thermostat. The circuit construction is quite simple, we have a voltage source and a resistor. Since the resistor is so large, there is very little voltage drop across it. Further, the next component has high input impedance, therefore the output of this stage is extremely close to the voltage generated by Vset. The voltage pattern specified by the PWL command can be seen in the following graph. It starts at 0V and jumps to 10V at 500ms, where it remains for the remainder of the simulation. This setpoint therefore instructs the PID controller that our desired state of the plant has changed.

Figure 1: PWL(0 0 500m 0 501m 10 5000m 10) Command Voltage Ouput

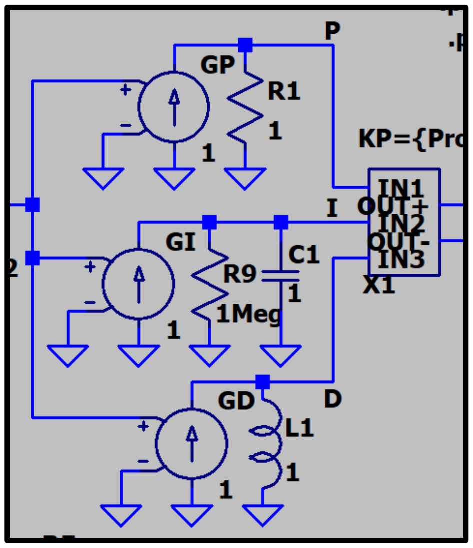


Error Generator



Given the setpoint voltage generated above, and a signal from a sensor which detects the current state of the system being controlled, we can generator an error signal. In many cases, this error signal is simply the difference between the two inputs. In our circuit, the sensor voltage at the negative terminal is subtracted from the setpoint voltage at the positive terminal.

PID Controller



Now, the most important part of a PID control system is the PID controller itself. As an input we have the error signal previously generated. This signal is fed into 3 different unit gain buffers. From there we must look at the proportional, integral, and derivative stages separately. At the end, however, they come back together into a summing amplifier which can tune the gain of the proportional, integral, and derivative stages separately. With the individual PID stages set, altering the summing amplifier coefficients is all we need to modify the response of our system. These coefficients are what will be modified later in the lab.

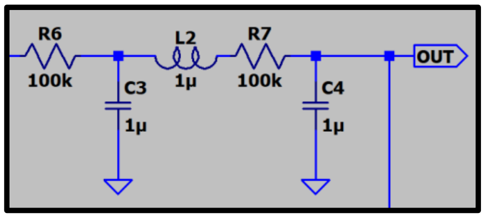
Looking at the proportional stage alone, we simply have another unitary gain stage. In this way, the first signal going into the summing amplifier is directly proportional to the original error signal generated.

The integral stage contains a low pass filter through a resistor and capacitor connected in parallel with ground. From our understanding of electronics, we know that a low pass filter is equivalent to an integrator. Therefore, the second signal going into the summing amplifier is proportional to the integral of the error signal.

Lastly, the derivative stage simply includes an inductor conducted to ground. We know that the voltage induced by an inductor is proportional to its self-inductance and the current flowing through it. Through some mathematical manipulation, we can show that the current flowing through the inductor is proportional to the rate of change of the input voltage. Therefore, as you would expect, the third input into the summing amplifier is proportional to the derivative of the error signal.

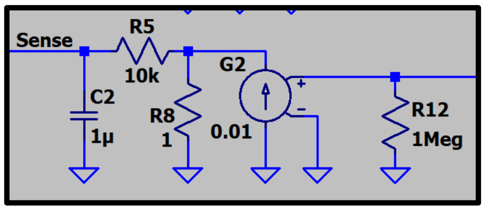
Plant

Simply put, the plant is the system being controlled. With the inclusion of capacitors and resistors, our plant will have both a time delay and inertia, making it harder to control and therefore incentivizing the need for a PID controller.



Sensor

Lastly, the sensor is the subcircuit used to identify the current state of the plant. In reality, the sensor will typically consist of some sort of transducer. However, in our case, it is simply a voltage divider in tandem with a low pass filter to eliminate high frequency noise. The output of this subcircuit will lead into the negative terminal of the error generator.



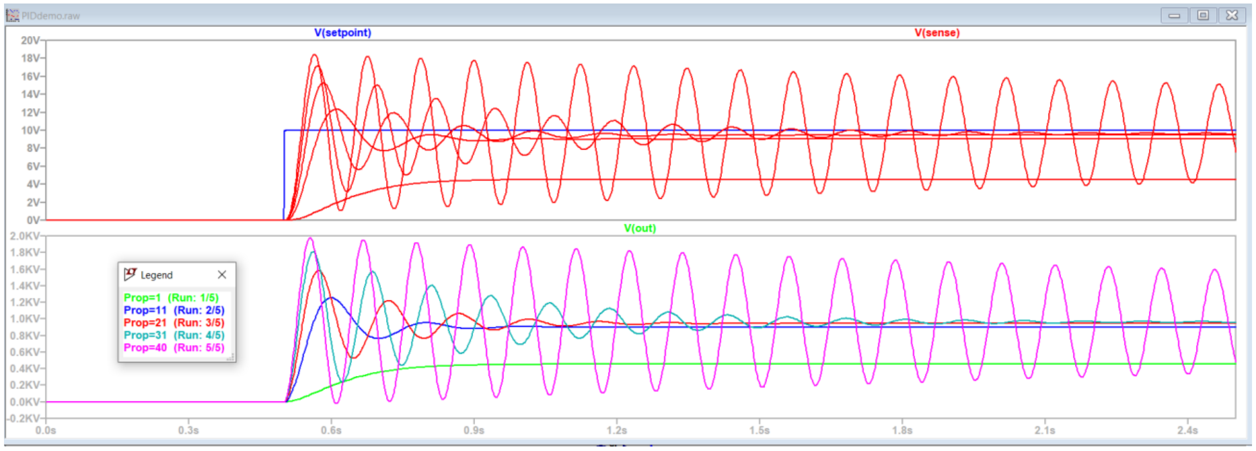
PID EXPLORATION

Using the PWL command specified by the setpoint voltage, we change the desired state of the plant. By probing the output voltage of our plant, we can see how it responds with time (driven the PID controller). In the following section we will first explore the effects of zeroing the gain for two PID parameters, while altering the gain for the third. We will then attempt to produce more complex controllers through various means.

For each of the following sections we will first present data gathered in the form of graphs generated through commands in LTspice. Following each graph will be an interpretation, highlighting the key aspects of each graph or explaining the process by which the graph is made. Procedurally, there is nothing more happening behind the scenes other than altering the gain parameters in LT spice and manipulating the outputted graphs.

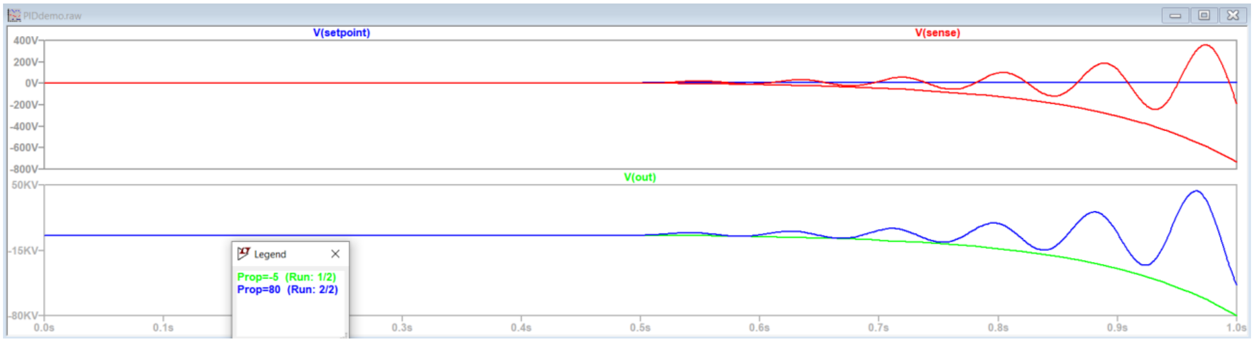
Proportional Gain

Figure 2: Voltage Outputs of Proportional Controller - .step param Prop 1 40 10



In the above graph we used the step command to alter the proportional gain from 1 to 40 in increments of 10, as seen in the legend. In this range we see a wide diversity of voltage outputs. Starting with the lowest gain of 1, we see that our output never reaches the desired value, it increases a bit and then maintains a steady state below our desired output. On the other extreme, when the gain equals 40, our output voltage oscillates greatly. It is evident that the amplitude of the output voltage is slowly decreasing. However, it will be quite some time before the output voltage can get close to reaching a steady state. With the intermediate values we see a gradual transition between these two extremes. One thing to note is that with higher proportional values, the eventual steady state voltage is closer to the desired output voltage. As can be seen in the top plot pane, where higher gain voltage outputs settle closer to the setpoint.

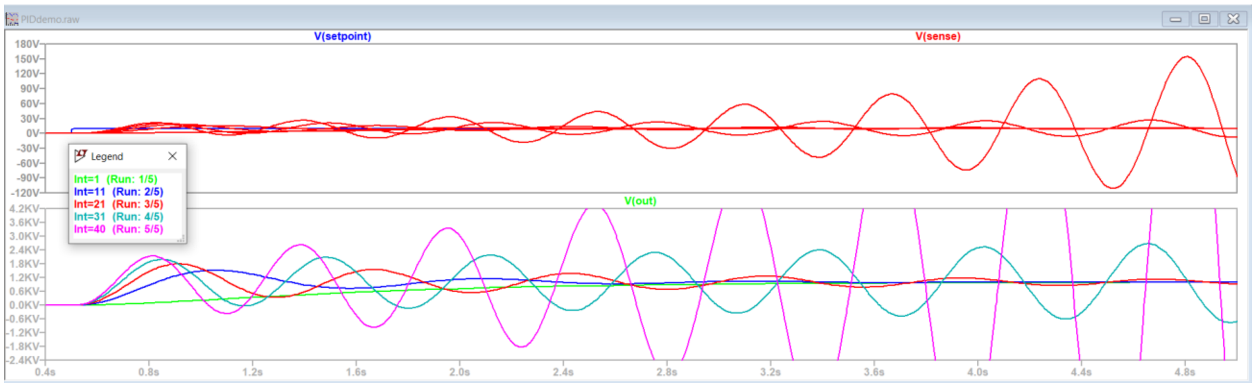
Figure 3: Voltage Outputs of Proportional Controller - .step param Prop -5 80 85



The increased accuracy in the steady state of course has its limits, as highlighted by the above figure. When the gain becomes too large, the amplitude of the sinusoidal behavior increases with time. At this point we have clearly lost control. Additionally, with a negative gain, rather than the voltage output being driven towards the desired output, it is driven away. Of course, in this scenario we also lose control.

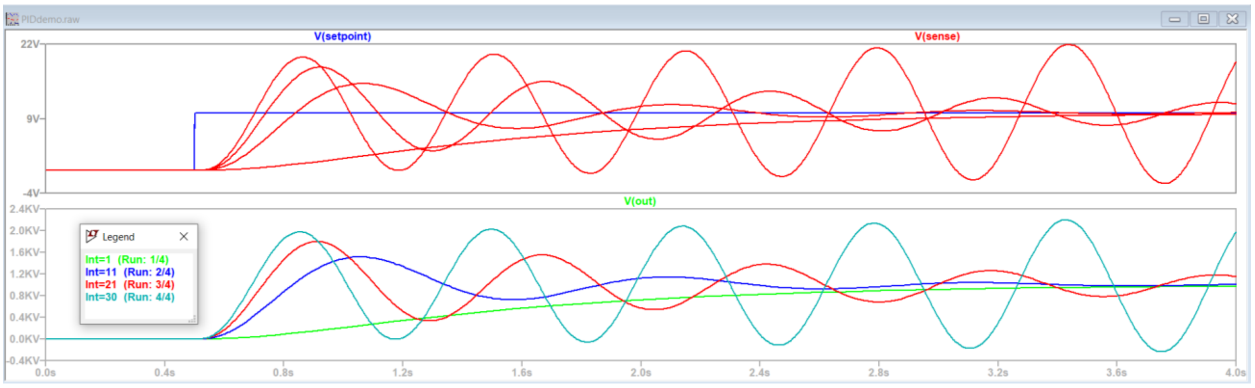
Integral Gain

Figure 4: Voltage Outputs of Integral Controller - .step param Int 1 40 10



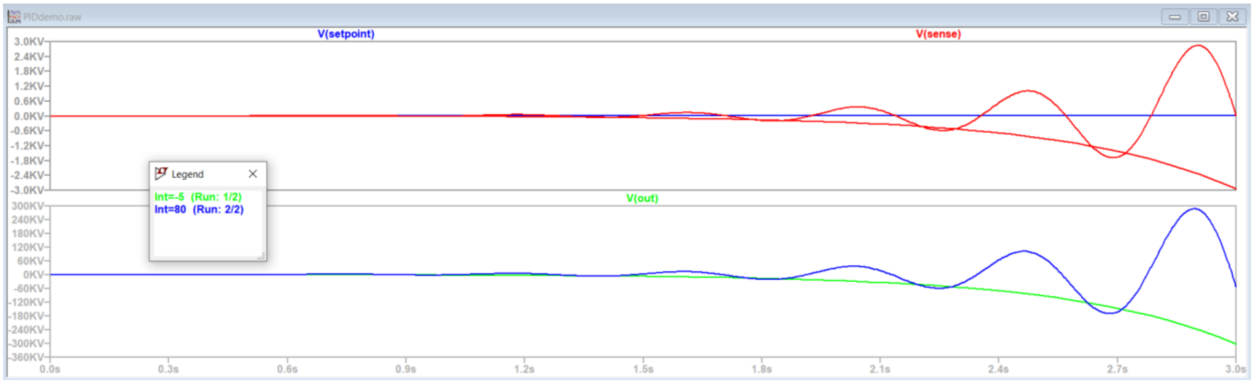
In the above graph we repeat the same step command as was done for the proportional gain, but this time with integral gain. We again see a range of behaviors. In the scale of the above figure, it is difficult to see the small gain characteristics. However, as with the proportional gain, if our integral gain becomes too large, then the oscillatory nature of the output increases amplitude with time.

Figure 5: Voltage Outputs of Integral Controller - .step param Int 1 30 10



In the above graph we restrict the upper limit of the step command to observe the output voltage features for smaller gain values. At the lower limit, we see no oscillatory behavior, the output voltage slowly approaches the desired value, never exceeding it. At our new upper limit, we again see (although more subtly) a loss of control. Finally, we see the most desirable output with the intermediate gain values. Namely, with a gain of 11, we see the output voltage come close to the desired value much quicker than the low gain scenario. Moreover, the oscillatory nature of the output due to overshooting is quickly dampened.

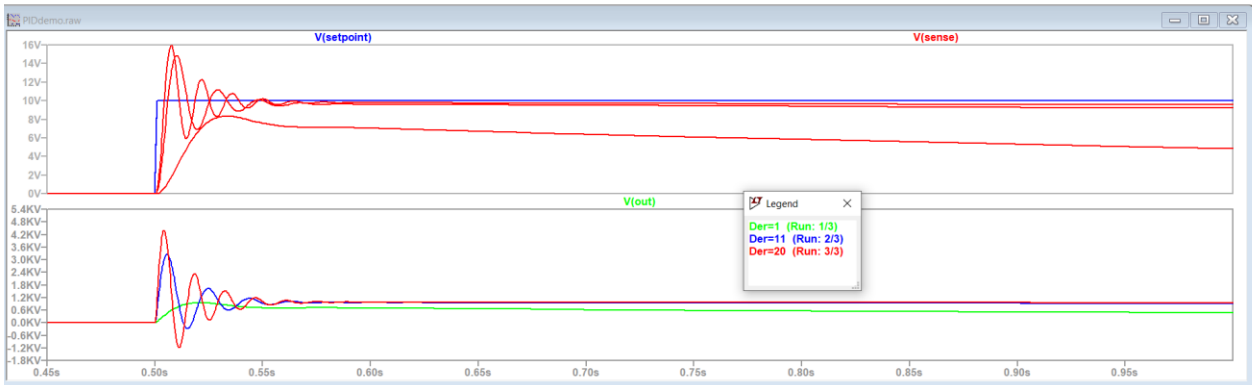
Figure 6: Voltage Outputs of Integral Controller - .step param Int -5 80 85



For this final experiment with an integral controller, we clearly show the effects of excessive gain driving the output voltage beyond control. Additionally, we show that with a negative integral gain, the output voltage is again driven away from the desired output.

Derivative Gain

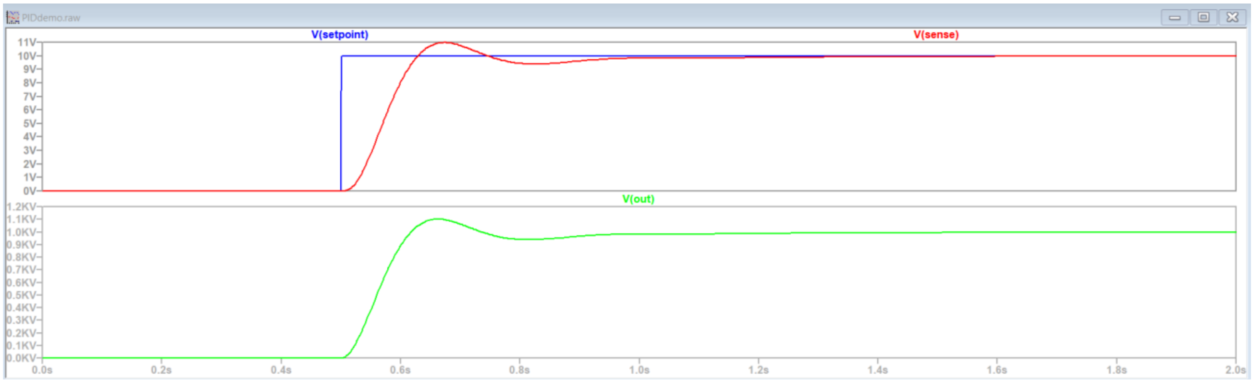
Figure 7: Voltage Outputs of Derivative Controller - .step param Der 1 20 10



Our last investigation of individual gain parameter examines the effects of using only derivative control. We manage to capture its range of behavior with only three gain values. When the gain is small, the output voltage responds quickly to the impulse by approaching the desired output. However, if we are at the desired output voltage, then the slope of our error will be 0, therefore eliminating the derivative term and causing our output voltage to deviate from the desired value. This drift is resisted by the derivative term but can not be eliminated with out infinite gain. We see with higher gain values, that the drift away from the setpoint voltage is slower. Additionally, with higher gain values, the derivative terms respond so strongly to the initial setpoint jump that it overshoots the output voltage. After some, time, they correct for their overshoots and are then subject to the previously mentioned drift away from the desired output. With higher gain values the over and undershoots are more severe. This should be taken into account because some systems would be sensitive to these extremes.

Optimizing PI

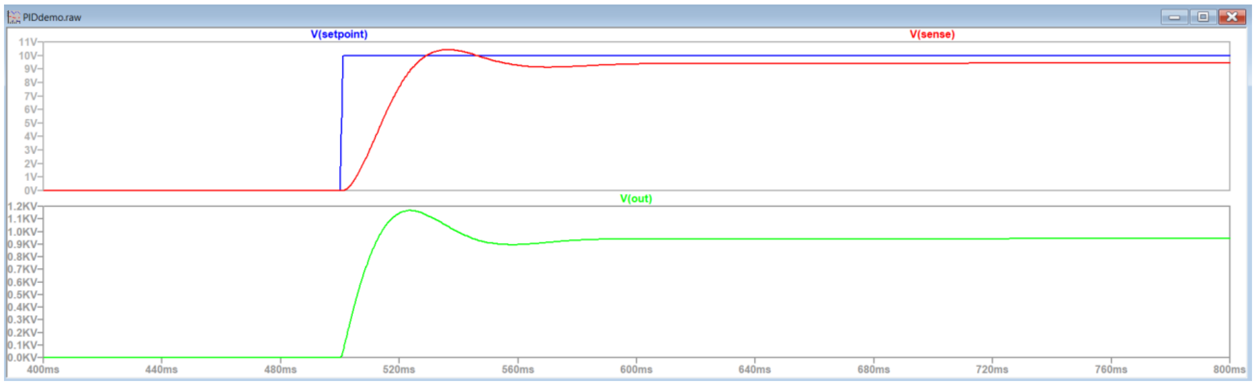
Figure 8: Voltage Outputs of PI Controller - P = 5, I = 15, D = 0



This response reaches 90% of the desired output in 0.10 seconds. The output voltage of the first extremum is 1.102kV, the second is 0.938kV, and the steady state voltage is 0.999kV. From the qualitative shape of the graph, we can see that the output reaches its desired value very quickly with minimal overshoot. The process for achieving these parameters was analytical. Given our observations from the effects of proportional and integral gain independently, we first set our proportional gain equal to 10. After testing nearby values, it is then decreased to 5 to reduce the magnitude of the oscillations and quickly reach a steady state. It is important to note that this steady state is not at the desired output, however, the proportional gain is designed to compensate for constant deviations from the set point. So, the next step is to add proportional gain. Again, following our conclusions from studying each gain parameter individually, we start with a value of 10. We then alter this parameter until we reach a desirable output. Out final output is somewhat subjective. For instance, we could further increase the proportional gain and reach our desired steady state output quicker. However, this comes at the cost of increasing the size of our overshoots. We chose for our output to stay within an approximate 10% error range when compared to our desired output once it comes within 10% for the first time.

Optimizing PD

Figure 9: Voltage Outputs of PD Controller - P = 20, I = 0, D = 1

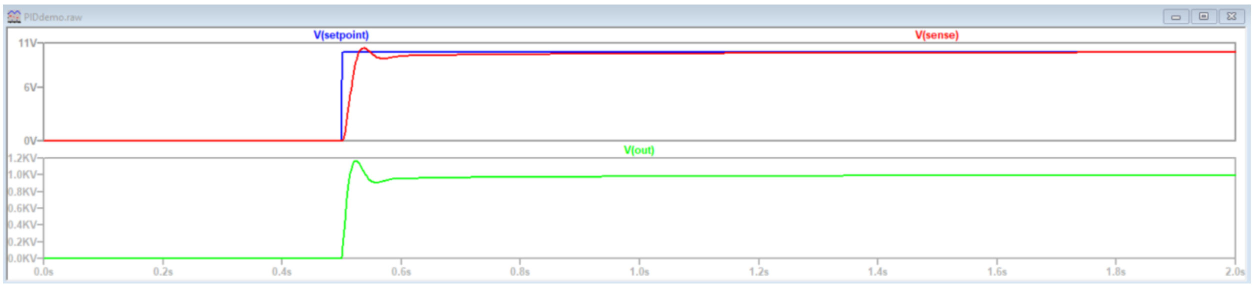


This response reaches 90% of the desired output in 0.01 seconds. The output voltage of the first extremum is 1.167kV, the second is 0.895kV, and the steady state voltage is 0.943kV. Again, from the qualitative shape of our graph, we have produced a very well-behaved PD controller. This design is rapid in its behavior, but not as precise at longer time intervals when compared to the PI controller. The exclusion of an integral gain seems to significantly effect the accuracy of the outputs steady state voltage. Furthermore, with its rapid nature, the two overshoots are also slightly more significant than in the PI controller. However, in picking these gain values, a key design choice is to minimize these overshoots. As with the PI controller, we start off by picking an approximate proportional gain. We again start with 10, leaving the other parameters at 0. Knowing that the derivative gain will counter the effects of the proportional gain, we increase the gain to 15 before adding a small derivative gain of 1. This value of 1 is chosen to prevent the excessive oscillations of higher gains shown in 7. For completeness, nearby values of derivative gain are also explored. However, with a value of just 2, the initial overshoot increases to greatly to 1.441kV. With a value of 3 the overshoot jumps to 1.739kV. When compared with the initial overshoot of 1.112kV using a derivative gain of 1, these elevated values are not acceptable. We now turn back to the proportional gain to determine if it can be optimized further. We encounter another tradeoff. The higher the proportional gain, the closer the steady-state output is to the desired value. But, this comes with additional initial overshoot. We therefore try to balance these two parameters. After testing multiple values, we arrive at a proportional gain of 20, giving us a first peak of 1.167kV and a steady state voltage of 0.943.

Comparing this final optimized PD controller with PI controller, we see that it responds much faster. This speed inevitably comes at the cost of having a less precise final output. These factors should be taken into consideration if choosing between the two controllers. The qualification of one controller being better than the other depends entirely on the plant's control requirements.

Optimizing PID

Figure 10: Voltage Outputs of PID Controller - P = 20, I = 30, D = 1



This response also reaches 90% of the desired output in 0.01 seconds. The output voltage of the first extremum is 1.170kV, the second is 0.908kV, and the steady state voltage is 0.996kV. Once again, from the qualitative shape of our graph, we have produced a very well-behaved PD controller. Visually, this graph looks very much like the PD control graph. The main difference is in the steady state behavior. Within 1.5 seconds the output is stable within 99% of the desired output, as is evident in the above graph. So, we have the rapid nature of the PD control in the early stages of the graph plus the accuracy of the PI controller in the later stages. Analytically, this was the desired output when picking parameters. Considering the derivate variable seems to be the most sensitive we start with the PD controller from the previous section and simply add the integral gain. We start with a modest value such as 10 and slowly increase observing the behavior. The tradeoff is again between overshooting and steady-state output. However, with any appreciable amount of integral gain the steady state approaches the desired value. It is therefore a decision of how quickly we reach that value and how significant our overshoot is. We again try to stay within the 10% error but do exceed that margin by a bit in exchange for being within a 1% of the desired output in 1.5 seconds. Attempts were also made to optimize the proportional gain; however, we found it to already be at the most desirable output.

Heuristic PID Optimization

In practice, we will rarely try to optimize the gain coefficients of our PID controller. Instead, we will use heuristic methods which have been developed to produce optimal gain coefficients. One of these methods is the Ziegler-Nichols rule. The Ziegler-Nichols rule uses two measured feedback loop parameters derived from measurements…

1. the period *Tu* of the oscillation frequency at the stability limit
2. the gain margin *Ku* for loop stability

…to calculate the three gain parameter with the following naming convention…

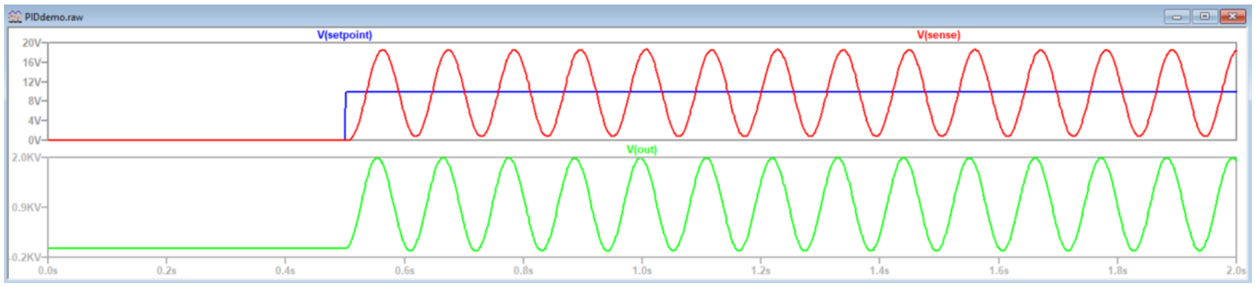
1. *Kp* - the controller path gain
2. *Ti* - the controller's integrator time constant
3. *Td* - the controller's derivative time constant

…where 1, 2, and 3 equate to what has previously been called proportional, integral, and derivative gain. To calculate K_p, T_i, and T_d from T_u and K_u we simply use the following table.

Rule Name	Tuning Parameters
Classic Ziegler-Nichols	Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu
Pessen Integral Rule	Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu
Some Overshoot	Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu
No Overshoot	Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

The stability limit occurs when both the integral and derivative gain are set to 0. We then change the proportional gain until our output responds sinusoidally with a constant amplitude.

Figure 10: Voltage Outputs at Stability Limit - P = 41, I = 0, D = 0



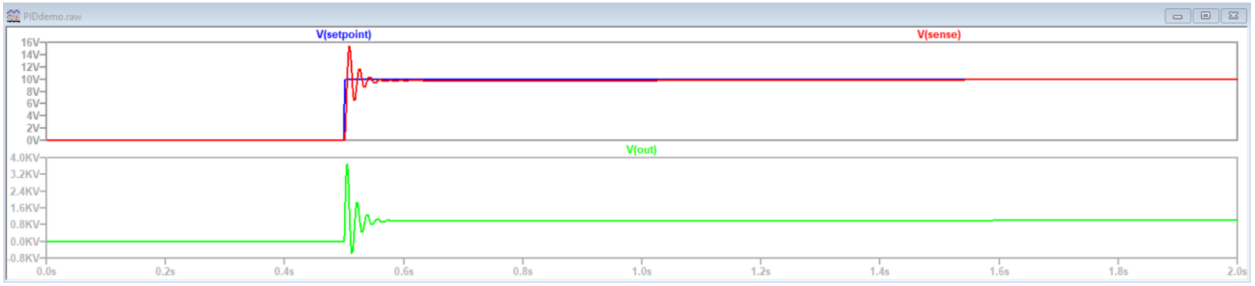
As shown in the title of the above figure, we found the stability limit for our particular circuit to be when the proportional gain, K_u, equals 41. Using the cursors in the above figure (not shown), we get a period of $T_u = 0.11s$.

Following the Classic Zeigler-Nichols rule and plugging our values into the table we get…

$$K_p = 25 \quad T_i = 55 \quad T_d = 14$$

…which produce the behavior displayed in the following figure.

Figure 11:Voltage Outputs of Ziegler-Nichols Tuned PID – P = 25, I = 55, D = 14

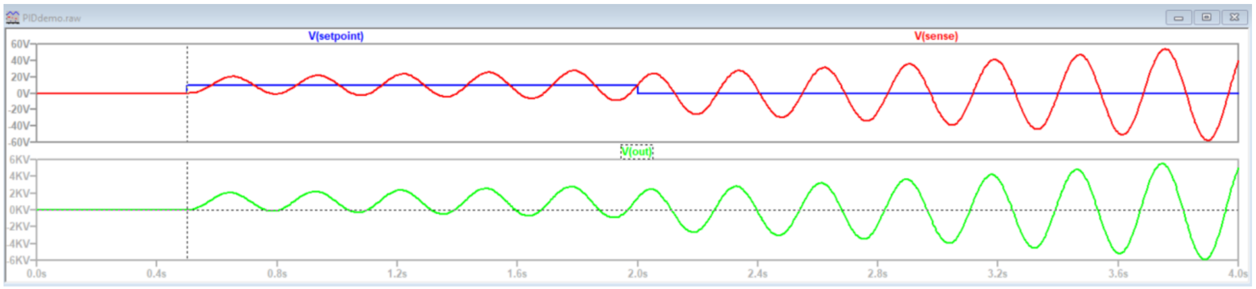


As we can see the above output reaches very good precision (less than 5% error) within 0.07 seconds. However, there are very strong oscillations prior to that. So large that the output voltage even drops below the starting point at the second extremum. If these oscillations are not a problem, then this gain configuration could be considered marginally better than our previously produced values. The controller reaches steady in approximately the same time and just barely achieves less than 1% error in 1.5 seconds. However, if these oscillations are burdensome then these parameters are not a good choice. It is quite impressive that given the two measurements we made, we were immediately able to produce such a good response. Further, there are additional rules shown in the table, such as "no overshoot" which could also produce great results, but for different applications.

Integral Windup

From a higher-level point of view, integral windup occurs when the integral gain saturates the output of a PI or PID controller. By saturation we mean that the final control element is driven to its limits. This is analogous to a valve being fully opened or closed. Once this limit is reached, there tends to be delay before the integral can become unsaturated. Consequently, the final control element remains saturated longer than it should. In the end, this results in a loss of control which can manifest itself as oscillations which increase in amplitude. To produce this effect, we return to the PI controller used in Figure 8, with a proportional gain of 5 and an integral gain of 15. We then increase the integral until we see a loss of control.

Figure 12: Voltage Outputs for Integral Windup – P = 5, I = 140, D = 0



To further accentuate the effects of integral windup we altered the PWL signal so that it returns to 0V at 2 seconds. Quite remarkably, the PI controller behaves in a controlled manner up until a proportional gain of around 140. At this point we start to see the effects of integral windup, characterized by the increased size in oscillations, driving us away from the desired output voltage.

CONCLUSION

PID controllers are quite remarkable pieces of equipment. Through these simulations in LTspice we have developed a better understanding of how to tune them appropriately. Furthermore, we now know the visual ques for the behaviors they exhibit both when operating properly and when malfunctioning. We managed to define the gain values for various controllers, all of which would provide tight control but with different trade-offs. In this manner, we can conclude that all PID controllers are created differently. The approach which one takes to tune a PID controller depends entirely on its applications and the desired behavior of the output.

SOURCES:

https://web.archive.org/web/20080616062648/http://controls.engin.umich.edu:80/wiki/index.php/PIDTuningClassical#Ziegler-Nichols_Method

https://controlguru.com/integral-reset-windup-jacketing-logic-and-the-velocity-pi-form/