How to Control a Process With Long Dead Time

by John Gerry, P.E.

Click here to see the complete presentation in more detail as a web-cast.

A process with a large dead time presents a special challenge for a controller - any controller. Find out how to get the best control of large dead time processes.

So Hard and the Simple Way

In a dead time process, the controller makes a change, then waits, waits, waits until the dead time has passed. Finally the controller finds out how its change effected the process variable. It is like trying to drive a car blindfolded with the passenger telling you what to do. You will need to go very slowly. The first try at getting better control in a dead time process is to reduce the dead time. Simply moving a probe closer to the valve may do it. But sometimes there is nothing you can do to lower dead time. So what can you do?

The advanced controller that is proven to work very well on dead time processes is a Proportional Integral controller with the combination of special stealth tuning parameters. It is important to not use derivative unless there is some lag or first order response in the process. Using derivative on a dead time process would make the control loop unstable.

Stealth Tuning

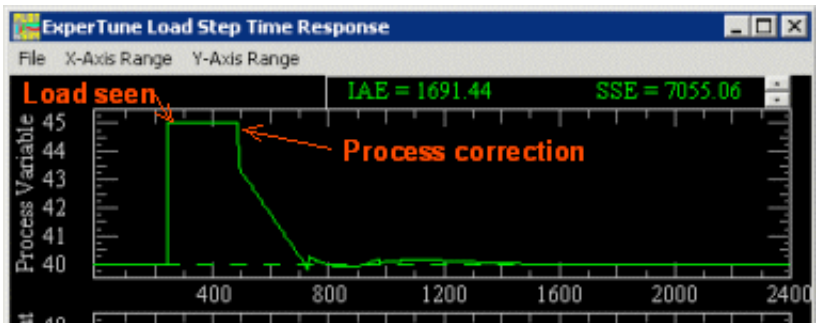
This stealth tuning applies to dead time processes. If the process has a lag, then this tuning still applies if the lag is 1/5 or less than the size of the dead time. The stealth tuning uses the process dead time and the process gain:

controller gain = 0.3/(process gain)
integral time = .42 * (process dead time)

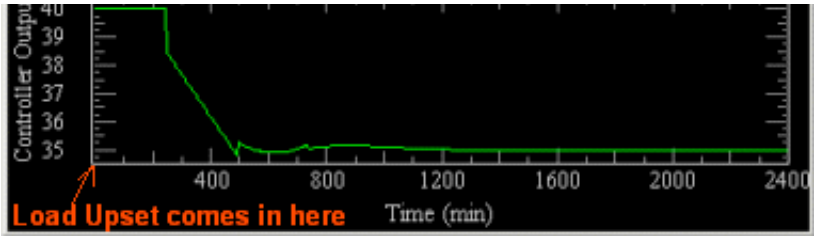
The integral units are in time - either minutes or seconds. These equations give optimal tuning -- minimum error to load upsets. If you want slower response, with more robustness, simply lower the gain some. (Note that these equations apply for usual ideal and series PI algorithms. For the parallel algorithm you've got to divide or multiply the integral time by the controller gain.) Controller algorithms are the subject of another presentation "Differences in PID Algorithms and Units" available from the articles page on ExperTune.com.

Example - Test of Stealth Tuning

To test our PI controller with stealth tuning, we have applied it on a process with 4 hours of dead time and a gain of 1. Here is the response to a load upset. The lower graph shows how the controller output moves to control the upset. The upper graph shows how the process variable responds.



The load comes in at time zero, but because of the dead time, the process variable doesn't see the upset until 240 minutes or 4 hours later. At this time, the controller responds, but another 240 minutes of dead time go by until the process variable responds to this controller action. Because of the dead time, this is the fastest possible response for ANY feedback controller. Another dead time passes until the process settles.

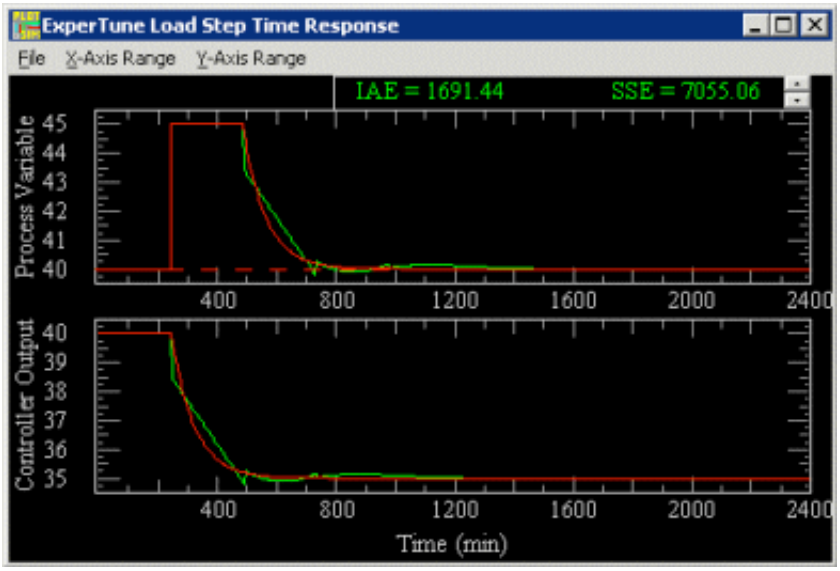


How to Make it Adaptive

To make our advanced dead time controller an adaptive one, you must be able to measure or infer the dead time from a variable like the speed of a machine or flow. Using the new dead time, re-calculate the new integral action and insert it into the controller. The key is to be able to measure or infer the dead time. This kind of adaptive controller is much more robust than using some kind of automatic identification technique. Attempting to identify the dead time using a least squares fit or some other modeling method that looks at normal process data is subject to gross modeling errors with disturbances. Identification techniques relying on setpoint changes, however, are accurate.

Controllers with Fancy Names

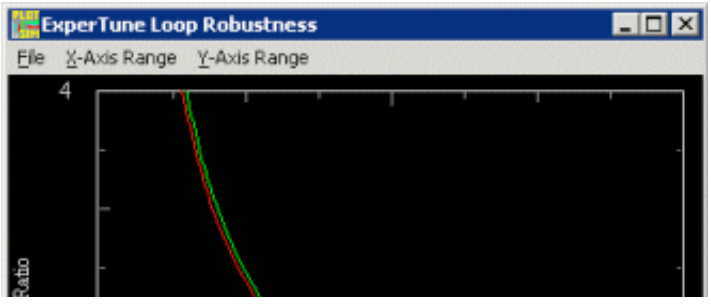
You can also use a fancy model predictive, model based, pole-cancellation, Internal Model Control, Dahlin, Direct Synthesis Controller or Smith Predictor. They all amount to the same thing of using a model inside the controller. Let's just call them all Model based controllers. Are these fancy controllers better than our PID controller with stealth tuning? Let's find out.



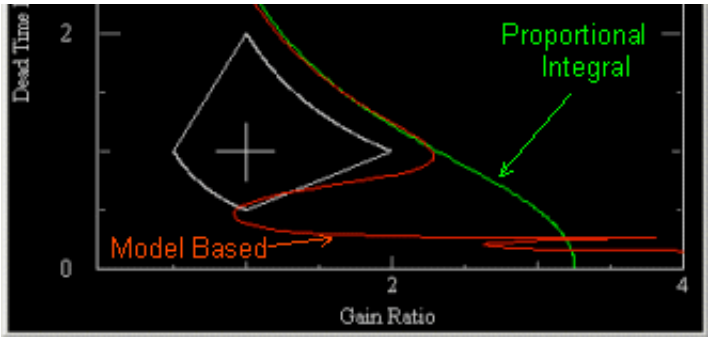
In red is the response of our model based controller compared to the PI. The response shows that model based controllers can eek out *slightly* better response than our PI controller - but this is at the sacrifice of robustness. Detailed discussion of robustness plots are the subject of another presentation "Robustness Plots - The Other Side of the PID Tuning Story" available on the articles page at ExperTune.com Let's see what the sacrifice is.

Fancy Names - Not So Fancy Results

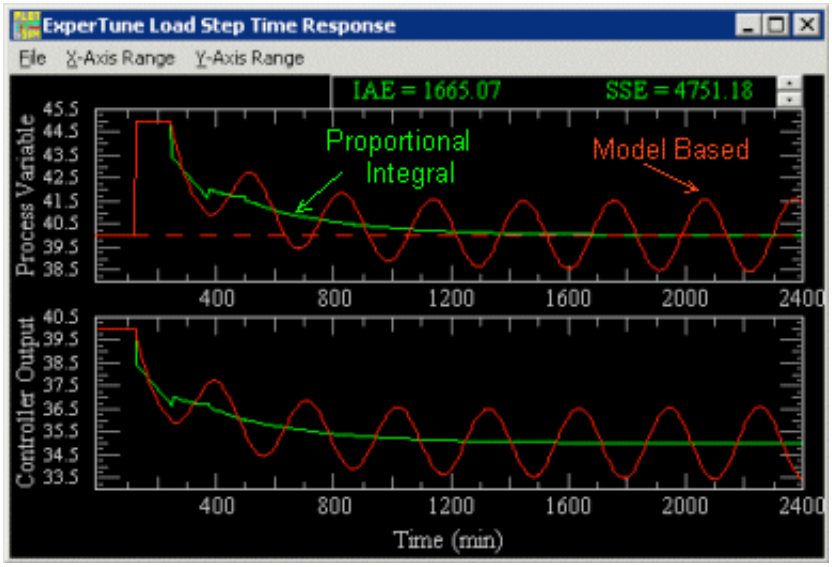
Here are the robustness plots for the PI and Model based controller in our example process. The green line is for our PI controller. The red line is for the model based controller. The robustness penalty with the model based control is severe -



the dead time of the process can go up or **down** and the loop will go unstable. Increasing dead time with any process will always eventually drive you unstable. But decreasing dead time to make you unstable is weirdly counter-intuitive. If the dead time were to go down with a PI controller, the loop would remain stable. Let's try this in our test process to see.



Here are the responses when we reduce the process dead time to 2 hours. The PI controller gets slower as expected. Consistent with the robustness plot, the Model Based controller goes unstable.



Summing it up

The best way to get better control of a dead time process is to reduce the dead time. A PI controller with proper tuning gives fast, stable response, and it can be adaptive. There are some other tricks that can help the response. For example applying a small filter to the process variable can smooth the response. Also if the process has a small lag, you CAN use a little derivative very carefully. For a process with a larger lag, using derivative can usually help response. Derivative, the Good The bad and the Ugly is the subject of another presentation available from the articles page at ExperTune.com.

ExperTune's PID Tuner software makes all of this easy and automatically adjusts its PI or PID tuning to work with your process whether it be mostly dead time, dead time with a lag, second order with dead time, integrating with dead time, or integrating with dead time and lag.

Click here to see the complete presentation in more detail as a web-cast.